# Mobil Uygulama Programlama

**Dr.Öğ.Üyesi Ömer ÇETİN**

**Konu-4**

## Bu Dersten Önce

- Mobil uygulama programlama nedir? Neden ihtiyaç duyarız?
- Web programlama ile farkları nedir? Ne gibi avantajlar sağlar?
- Android uygulaması nedir?
- Android uygulama geliştirme ortamları nelerdir? Nasıl sağlanır?
- Android Studio IDE'sinin tanıtılması ve ilk proje geliştirilmesi ve derlenmesi
- Android Studio ile geliştirilen projenin bir mobil cihaz ve sanal makine üzerinde çalıştırılması
- Uygulama konfigürasyonu ve manifest dosyası
- Aktiviteler, aktivite durumları, yeni aktivite oluşturma
- Intent, Service, Broadcast, Content Providers, Resources (kaynaklar)

# Bu Derste

- Gradle
- Ara yüz geliştirme
- Android layout türleri
  - Linear Layout
  - Relative Layout
  - Frame Layout
  - Grid Layout
- Dinamik Android Layout Oluşturma



# Gradle

- Gradle, Android uygulaması geliştirme aşamalarını otomatize eden açık kaynak kodlu **Android Studio üzerinde çalışan bir yapı sistemi**dir.
- Android Studio da bir proje oluşturduğumuzda **otomatik olarak gradle build sistemi** devreye girer ve **build işlemi**ni gerçekleştirir.
- Android Studio da bir Android projesinin genel yapısını önceki derslerimizde inceledik. Bildiğiniz üzere bir Android projesi bir den fazla modülden oluşur. Projeyi test edebileceğimiz, hataları ayıklayabileceğimiz, oluşturduğumuz uygulamayı yayınlayabilmek için apk oluşturabileceğimiz bir çok bileşenden meydana gelen bu yapıda aslında **her bir bileşenin kendine ait bir build sistemi** vardır.

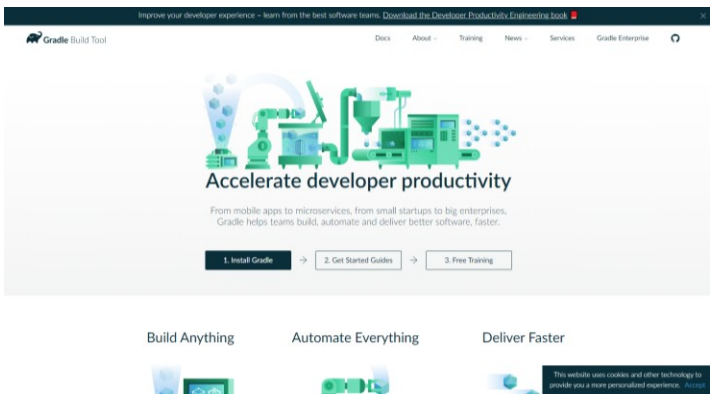
# Gradle

- Genel olarak Android projelerini incelediğimizde **iki tür gradle oluşturma** dosyası yer aldığı görülür.
- Eclipse ile Android uygulaması geliştirirken gradle yoktur yerine **maven yapısı** vardır ve maven projelerinde sadece .jar çıktıları kullanılabilirken **gradle ile birlikte xml çıktıları, fontlar vb. yapıları içeren .aar çıktıları**nı da kullanabilirsiniz.
- Android Studio'da yeni bir Android projesi oluşturduğumuzda proje oluşturulma aşamasında gradle çalışıyor ve build işlemlerini tamamlıyor.

Gradle

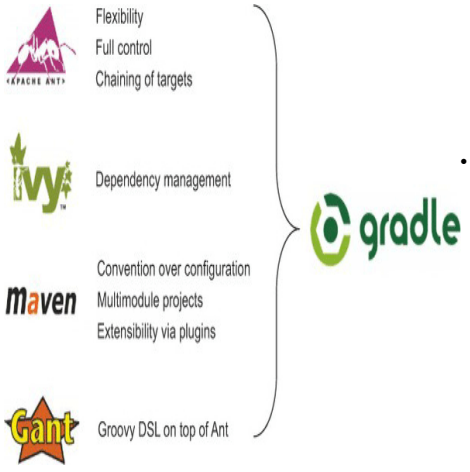


# Gradle




https://gradle.org/

# Gradle




- Ant, Maven veya Ivy'nin her birinin kendilerine ait güçlü yanları ve zayıf yanları mevcut. Gradle projesinin farkı ise işte bu **diğer inşa araçlarının güçlü yanlarını bir araya getirmesi**.
- Gradle'ın sağladığı en büyük avantaj **kütüphane yönetim sistemi**dir. Projenizin ihtiyaç duyduğu herhangi bir açık kaynak kodlu kütüphaneyi, projenize tek satır kod ile ekleyebilirsiniz. (**app** dizinindeki **build.gradle** içerisinde bulunan **dependencies** içerisinde bu işlemi kolayca gerçekleştirebilirsiniz böylece kullanacağınız kütüphaneyi otomatik olarak internet üzerinden sizin için indirecektir.)



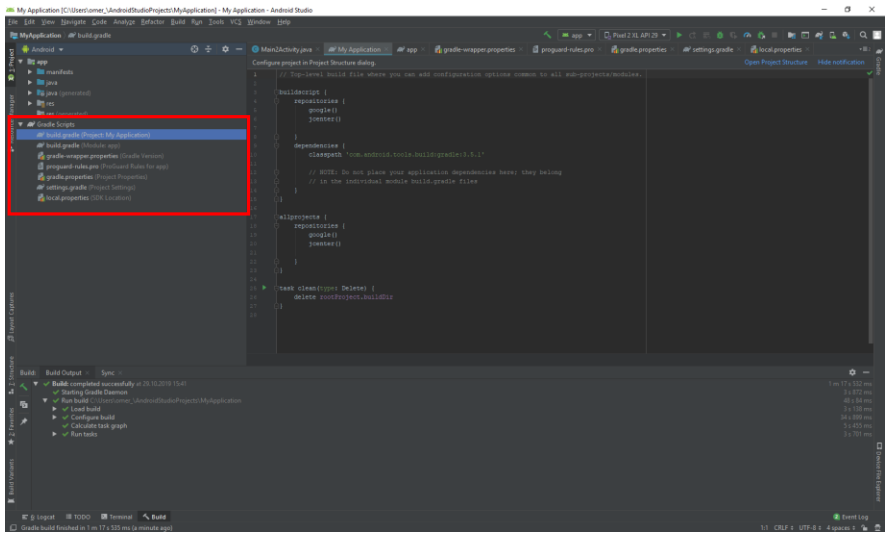
# Gradle


- Örnek verecek olursak Picasso kütüphanesini projemize dahil etmek istiyoruz yapmamız gereken tek şey aşağıdaki kod parçacığını anlattıldığı gibi dependencies içerisine yapıştırmaktır.

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:19.+'
}
```

- Bu kısımdaki ilk satır, **libs klasöründeki .jar dosyalarının projeyle beraber compile edilmesi gerektiğini** belirtir. İkinci satır ise **Maven Central'da bulunan bir repository'yi kaynak olarak kullandığını** belirtir. Bunlar zaten siz projeyi oluşturduğunuzda kendiliğinden ayarlanmış vaziyette gelir.

# Gradle





# build.gradle (project)

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.

buildscript {
    repositories {
        google()
        jcenter()

    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.5.1'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()

    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

# build.gradle (app)

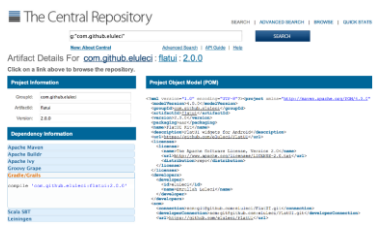
```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.2"
    defaultConfig {
        applicationId "com.example.myapplication"
        minSdkVersion 20
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    implementation 'com.google.android.material:material:1.0.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
}
```



# Gradle - Repository kullanımı

- Gradle'ın en büyük avantajı **projenizin bağımlı olduğu kaynakları internet üzerinden otomatik olarak indirebilmesi**dir. Eğer kullanmak istediğiniz kütüphanenin **remote repository**'si varsa tek satır kod ile kütüphaneyi projenize ekleyebilirsiniz. Bu repository bilgilerini kullanmak istediğiniz kütüphanenin sayfasından, **search.maven.org** veya **mvnrepository.com**'dan bulmalısınız.
- Örneğin search.maven.org 'a girip 'flatui' yazarsanız karşınıza Android FlatUI kütüphanesi çıkar. Repository sayfasına geldiğinizde ise sol alt tarafta Gradle/Grails sekmesine tıklarsanız kütüphaneyi kullanmanız için ihtiyacınız olan satırı görebilirsiniz.

compile 'com.github.eluleci:flatui:2.0.0'




```
dependencies {
    compile 'com.android.support:appcompat-v7:19.0.1'
    A newer version of com.android.support:appcompat-v7 than 19.0.1 is available: 19.1.0
    compile 'com.squareup.retrofit:retrofit:1.5.0'
    compile "com.google.android.gms:play-services:4.3.23"
    compile fileTree(dir: 'libs', include: ['*.jar'])
}
```

Bu resimde gösterilen dosya bir projenin build.gradle dosyası. İçinde projenin bağımlı olduğu kaynakları görebilirsiniz. Ayrıca burada belirttiğine göre kütüphanelerden birisinin daha yeni versiyonu çıkmış ve **Android Studio da güncelleyebileceğini** söylüyor.

Android için kullanılan ve yaygın olan çoğu kütüphane çoktan Gradle'a uyumlu hale gelmiştir. Bu yüzden bulduğunuz kütüphanelerin çoğunu rahatlıkla Gradle projenizde kullanabilirsiniz.

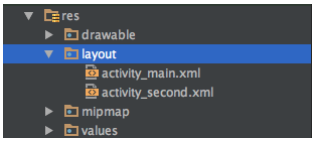
# Ara Yüz Geliştirme

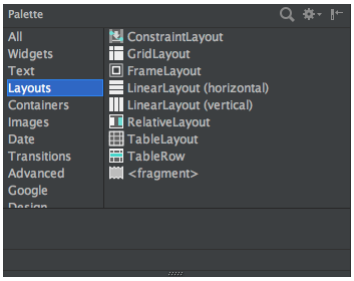
- Arayüze sahip yazılım projelerinde (Web Yazılım,Ios ve Android Yazılımlar vb.), projenin en az back-end yazılım bölümü kadar **ara yüz yazılım (front-end)** kısmının da düzgün hazırlanması büyük önem arz etmektedir.
- Ara yüz yazılımının doğru yapılandırılabilmesinin en önemli şartlarından biri, **tasarım taslağı** ile ara yüz yazılımının sonucunda çıkarılacak ara yüzün aynı olmasıdır. Bu noktada ara yüz yazılımcıları en çok **elementlerin boyutlandırılması**, **birbirleri arasında hizalandırılmaları** gibi konularda çok sıkıntı yaşarlar.
- Bir diğer önemli konuda ara yüzlerin değişken olarak bir başka değiş ile **dinamik** olarak tasarlanabilmesidir.



# Ara Yüz Geliştirme

- Android uygulamalarımızda görsel kısımlar **res klasörü altında bulunan layout dosyaları** tarafından oluşturulur ve bu oluşan bu dosyalar **xml formatında**dır. Android Studio da layout kısmını açtığınızda karşınıza gelen ekranda istediğiniz tasarımları yapabilirsiniz, Android'in sahip olduğu görsel bileşenleri ister kod ile (XML tarafında) isterseniz sürükle bırak ile oluşturabilirsiniz.




- Yanda gördüğünüz gibi uygulamada kullanabileceğimiz **çeşitli layout türleri** vardır. (listede fragment yapısına da yer verilmiş fakat farklı bir kavram olduğu için fragmentları layout türlerinde anlatılmayacaktır.)

# Android Layout Türleri

- Her bir layout **width** ve **height** özelliklerine sahiptir.
  - width yatay alanı , height ise dikey alanı belli eder.
- **match_parent**: Bulunduğu alan içerisinde yatay veya dikey olarak **yer aldığı alanı kaplar**. Bu özellik; **fill_parent** ile kullanılıyordu genelde ama android 2.2 versiyonundan sonra artık match_parent kullanılmaya başlandı.
- **wrap_content:** Bulunduğu alan içerisinde yer alan metin, resim veya **bileşen kadar yer kaplar**.

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
```



# Android Layout Türleri

- **RelativeLayout**: RelativeLayout ile birlikte Android görsel bileşenlerini istediğimiz yere sürükleyip kullanabiliriz. Bu layout içerisinde yer alan bileşenleri de kendi içerisinde konumlandırabiliriz sağına göre, soluna göre, alt kısma göre, üst kısma göre. Relative Layout ile daha esnek tasarımlar yapabiliriz.
- **LinearLayout**: LinearLayout ile birlikte tüm Android bileşenlerini tek bir konumda kullanabiliriz. Orientation özelliğini kullanarak yatay **LinearLayout (horizontal)** ve dikey **LinearLayout (vertical)** olarak da konumlandırabiliriz. Alt alta yada yan yana aynı çizgi üzerinde bulunacak componentleri LinearLayout ile kullanırız.

# Android Layout Türleri

- **TableLayout**: TableLayout ile birlikte Android bileşenlerini bir tablo yapısına yerleştirirsiniz. Html'deki gibi tablo yapısı şeklinde göstermek için kullanılır.
- **FrameLayout**: FrameLayout ile birlikte Android bileşenleri üst üste biner. Örnek verecek olursak; iki butonunuz var ve aynı yerde konumlanmasını istiyorsunuz, biri gözüktüğünde diğeri kaybolsun ve tam tersi koşulda gerçekleşsin istiyorsanız FrameLayout kullanabilirsiniz. Başka bir örnek ise elinizde bulunan bir Listview üzerinde bir butonun her zaman sabit kalmasını istiyorsunuz. Bu durumda FramelLayout kullanabilirsiniz.



# Android Layout Türleri

- **GridLayout**: **row** ve **column** yapısına sahip olan Android bileşenlerini grid'de toplayan layout türüdür.
- **ConstraintLayout**: Android Studio 2.2 ile gelen bu layout ile Android bileşenlerini sürükle bırak yaparak responsive bir görüntü elde edebiliriz ayrıca alt alta ve yan yana android bileşenlerini elde etmek için kullandığımız LinearLayout yerine ConstraintLayout kullanarak her cihaz için uyumlu tasarımlar oluşturabiliriz.

# Android Layout Türleri

1. **LinearLayout**
2. RelativeLayout
3. FrameLayout
4. GridLayout
5. TableLayout
6. ConstraintLayout



# LinearLayout

- LinearLayout aslında iki koşulda kullanma ihtiyacı duyuyoruz;
  - Birincisi Android bileşenlerini **yatay konumlandırmak** için;
  - ikincisi de Android bileşenlerini **dikey konumlandırabilmek** için.
  - Bu özelliği de **orientation** ile sağlıyoruz.

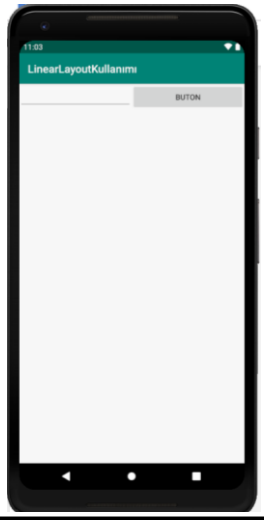
# LinearLayout

Aşağıdaki XML yapısında gördüğünüz gibi **android:orientation = "horizontal"** dediğimizde yatay olarak EditText ve Buttonu gösterdik.

**activity_main.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="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:orientation="horizontal"
        android:id="@+id/linearLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <EditText
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <Button
            android:text="Buton"
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />


    </LinearLayout>

</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN




# LinearLayout

Aşağıdaki XML yapısında gördüğünüz gibi **android:orientation = "vertical"** dediğimizde yatay olarak EditText ve Buttonu gösterdik.

**activity_main.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="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:orientation="vertical"
        android:id="@+id/linearLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <EditText
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <Button
            android:text="Buton"
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />


    </LinearLayout>

</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN

# LinearLayout

LinearLayout u **android:layout_centerInParent = "true"** ile cihaza göre tam ortaladık ve **android:background** ile de arka plana renk kodu verdik böylece LinearLayout un kapladığı kısım yeşil oldu.

**activity_main.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="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:background="@color/colorPrimary"
        android:layout_centerInParent="true"
        android:orientation="vertical"
        android:id="@+id/linearLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <EditText
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <Button
            android:text="Buton"
            android:layout_weight="1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />


    </LinearLayout>
</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN

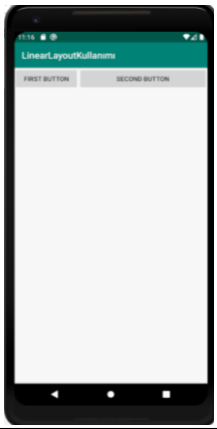

# LinearLayout

Aşağıdaki kod yapısına baktığımızda LinearLayout da **android:weightSum** kavramını kullandık ve **android:weightSum = "3"** dedikten sonra kapladığı alanda yatay olarak **3 birim yer** ayırmış oldu içerisinde yer alan butonların birine android:layout_weight="1" , diğerine ise android:layout_weight="2" olarak konumlandırdık. **Ekran görüntüsüne de bakacak olursak; 3 birimlik alanda 2 birim ve 1 birim alan butonların farklı boyutlarda konumlandığını göreceksiniz.** weightSum değerine kaç veriyorsak o değere göre bileşenleri sabitlemek önemli. Hem performans açısından hem de konumlandırmayı weight değerine göre düzgün bir şekilde yapabilmek için butonların **width değerlerine 0 dp** verdik.

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:weightSum="3"
    android:orientation="horizontal">


    <Button
        android:id="@+id/btn_blue"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="First Button" />

    <Button
        android:id="@+id/btn_green"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="Second Button" />
</LinearLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN



# LinearLayout

**android:gravity=”center”** kullanarak en dışta bulunan LinearLayout’umuzu ekrana göre ortaladık, böylece içinde bulunan butonlarda ekranın tam ortasında konumlanmış oldu.

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="horizontal">


    <Button
        android:id="@+id/btn_blue"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="First Button" />

    <Button
        android:id="@+id/btn_green"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="Second Button" />

</LinearLayout>
```




```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#c56504"
    android:padding="10dp" >

    <TextView
        android:id="@+id/emailTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="5dp"
        android:text="Email :"
        android:textColor="#FFFFFF"
        android:textSize="18sp" />
    <EditText
        android:id="@+id/emailEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email adresinizi girin"
        android:inputType="textEmailAddress">
    </EditText>
    <EditText
        android:id="@+id/passwordEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Şifrenizi girin"
        android:inputType="textVisiblePassword">
    </EditText>
    <Button
        android:id="@+id/saveButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Giriş" />
    <LinearLayout
        android:weightSum="2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
        <Button
            android:id="@+id/createAccountButton"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Yeni Hesap" />
        <Button
            android:id="@+id/forgetPasswordButton"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Şifremi unuttum" />
    </LinearLayout>
</LinearLayout>
```

**activity_main.xml**

Yandaki XML yapısına baktığımızda ise LinearLayout’ları **yatay ve dikey olarak birlikte** kullanabildiğimizi görüyoruz. En dıştaki LinearLayout dikey bir şekilde konumlandırdık. Alt kısmındaki iki buton’un yatay olarak konumlandığı yerde ise diğer LinearLayoutu kullandık.

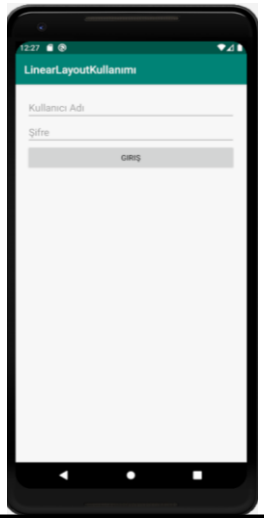
# LinearLayout

Aşağıdaki layout yapısına baktığımızda LinearLayout'a **android:layout_marginLeft**, **android:layout_marginRight**, **android:layout_marginTop**, **android:layout_marginBottom** değerlerine 20dp verdik, yani 20dp olarak sağdan, soldan, aşağıdan ve yukarıdan boşluk bırakmış olduk. **android:layout_margin = "20dp"** yaparak da bu dediğimiz özelliği sağlayabilirdik diğer değerleri silip sadece margin değeri verip kendinizde görebilirsiniz.

activity_main.xml

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:orientation="vertical"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="20dp"
        android:layout_marginRight="20dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <EditText
            android:hint="Kullanıcı Adı"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <EditText
            android:hint="Şifre"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <Button
            android:text="Giriş"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />


    </LinearLayout>
</RelativeLayout>
```






# Android Layout Türleri

1. LinearLayout
2. **RelativeLayout**
3. FrameLayout
4. GridLayout
5. TableLayout
6. ConstraintLayout

# RelativeLayout

- Linear Layout'da yatay ve dikey olarak konumlandırma varken, bu layout türümüzde böyle hazır bir şablon bulunmamaktadır. Relative Layout içine eklediğimiz **Android bileşenlerini birbirlerinin aşağısına, yukarısına, sağına ve soluna şeklinde konumlandırma** yapabiliriz. Ayrıca, birden fazla iç içe layoutları kullandığımız da ilgili layout'un id'sini belirterek, belirli methodlara göre hizalama yapabiliriz.



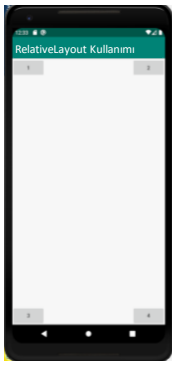
# RelativeLayout

**activity_main.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="match_parent">

    <Button
        android:text="1"
        android:layout_alignParentLeft="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="2"
        android:layout_alignParentRight="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="3"
        android:layout_alignParentLeft="true"
        android:layout_alignParentBottom="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />


    <Button
        android:text="4"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</RelativeLayout>
```



- Aşağıdaki xml layout yapısına baktığımızda Butonları android cihazın sağ üst köşe, sol üst köşe, sol alt köşe ve sağ alt köşeye göre konumlandırdık.
  **android:layout_alignParentLeft = "true"** sol tarafa göre konumlandırma, **android:layout_alignParentRight = "true"** sağ tarafa göre konumlandırma, **android:layout_alignParentBottom="true"** ile de en alt kısıma göre konumlandırmasını yaptık.



# RelativeLayout

**activity_main.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="match_parent">

    <RelativeLayout
        android:layout_centerVertical="true"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <EditText
            android:id="@+id/editUserName"
            android:hint="Kullanıcı Adı Giriniz"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <EditText
            android:id="@+id/editUserPassword"
            android:hint="Şifre Giriniz"
            android:layout_below="@+id/editUserName"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />

        <Button
            android:id="@+id/loginBttn"
            android:layout_centerInParent="true"
            android:layout_below="@+id/editUserPassword"
            android:text="Giriş"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

    </RelativeLayout>
</RelativeLayout>
```

- Aşağıdaki XML layout yapısına baktığımızda Liner Layout'da orientation'a dikey konumlandırma vererek yapabilceğimiz işlemi Relative Layout kullanarak android bileşenlerini "**android:layout_below**" ile alt alta konumlandırdık. Her bir bileşene kendine özgü id verdik ve editUserName altında editUserPassword olsun, editUserPassword un altında da loginBttn şeklinde belirledik.

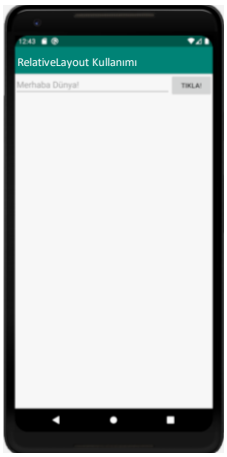



# RelativeLayout

- Aşağıdaki XML layout yapısına baktığımızda; EditText ile Button bileşenlerini yan yana konumlandırdık. EditText de **android:layout_toLeftOf="@+id/clickBtn"** ile id'si clickBtn olan buton nesnesinin solunda konumlandırma yaptık.

**activity_main.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="match_parent">

    <EditText
        android:id="@+id/writeEditTxt"
        android:hint="Merhaba Dünya!"
        android:layout_alignParentLeft="true"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@+id/clickBtn" />
    <Button
        android:id="@+id/clickBtn"
        android:text="Tıkla!"
        android:layout_width="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_height="wrap_content"/>

</RelativeLayout>
```



# Dinamik Android RelativeLayout Oluşturma

- Yukarıda XML layout kısmında oluşturduğumuz yapıyı **dinamik olarak** da oluşturabiliriz. İlgili Activity'mizin **onCreate methodunda setContentView'**e direk olarak XML layout dosyasını vermeyiz ve yukarıdaki işlemleri dinamik bir şekilde **onCreate** içinde kendimiz yazabiliriz.
- Aşağıda gördüğünüz gibi **setContentView**'i kendimiz oluşturacağımız için commente aldık.
- Öncelikle bir Buton yarattık ve **RelativeLayout.LayoutParams** ile butonun **width (WRAP_CONTENT)** ve **height (WRAP_CONTENT)** değerlerini oluşturduk.
- Yukarıda belirttiğimiz **android:layout_alignParentRight="true"** yerine **buttonParams.addRule(RelativeLayout.ALIGN_PARENT_RIGHT,RelativeLayout.TRUE)** olarak kural verdik.
- Butonun text değerini "TIKLA!" olarak set ettik ayrıca butona id değeri olarak 999 verdik. (Kod da belirttiğimiz bir değer bu siz istediğiniz int değerini verebilirsiniz.)

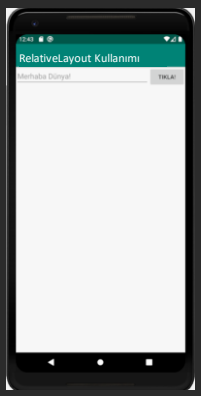


# Dinamik Android RelativeLayout Oluşturma

- Daha sonra EditText oluşturduk ve **RelativeLayout.LayoutParams** ile EditText'in **width** ve **height** değerlerini ayarladık (**MATCH_PARENT, WRAP_CONTENT**) daha sonra oluşturduğumuz bu params değerine kural verdik.
- Yukarıda belirttiğimiz **android:layout_alignParentLeft="true"** değerinin karşılığı olan **(editTextParams.addRule(RelativeLayout.ALIGN_PARENT_LEFT, RelativeLayout.TRUE))** değerini kullandık ayrıca **editTextParams.addRule(RelativeLayout.LEFT_OF, 999)** dediğimizde yine yukarıda belirttiğimiz **android:layout_toLeftOf="@+id/clickBtn"** değerinin karşılığı olan id'si 999 olan butonun soluna göre konumlandır dedik.
- Son olarak en dışta kullanacağımız RelativeLayout oluşturduk ve EditText ve Button bileşenlerini addView ile bu layout a ekledik. setContentView fonksiyonuna da oluşturduğumuz RelativeLayout bileşenini verdik böylece aşağıdaki kodu çalıştırdığımızda yine yukarıda elde ettiğimiz ekran çıktısının aynısını alacağız tek fark bu işlemlerin hepsinin dinamik olarak oluşturduk.

main_activity.java

```
package com.example.linearlayoutkullanm;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main);

        Button clickButton = new Button(this);
        RelativeLayout.LayoutParams buttonParams = new
RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.WRAP_CONTENT,
                RelativeLayout.LayoutParams.WRAP_CONTENT);
        buttonParams.addRule(RelativeLayout.ALIGN_PARENT_RIGHT,RelativeLayout.TRUE);
        clickButton.setLayoutParams(buttonParams);
        clickButton.setText("TIKLA!");
        clickButton.setId(999);

        EditText editText = new EditText(this);
        RelativeLayout.LayoutParams editTextParams =
                new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.MATCH_PARENT,
                        RelativeLayout.LayoutParams.WRAP_CONTENT);
        editTextParams.addRule(RelativeLayout.ALIGN_PARENT_LEFT, RelativeLayout.TRUE);
        editTextParams.addRule(RelativeLayout.LEFT_OF, 999);
        editText.setHint("Merhaba Dünya!");
        editText.setLayoutParams(editTextParams);


        RelativeLayout outsideLayout = new RelativeLayout(this);
        outsideLayout.setLayoutParams(new RelativeLayout.LayoutParams(RelativeLayout.LayoutParams.MATCH_PARENT,
RelativeLayout.LayoutParams.MATCH_PARENT));
        outsideLayout.addView(editText);
        outsideLayout.addView(clickButton);
        setContentView(outsideLayout);
    }
}
```




Dr.Öğ.Üyesi Ömer ÇETİN

# Dinamik Android RelativeLayout Oluşturma

- Aşağıdaki xml layout yapısına baktığımızda; en **ortada** ImageButton'u (**android:layout_centerInParent="true"**) kullanarak konumlandırdık. ImageButton nun sağına, soluna, aşağısına ve yukarısına ise Button ları konumlandırdık.
- **android:layout_toLeftOf="@+id/imageBttn"** : verilen id nin **solunda** kendini konumlandırır.
- **android:layout_toRightOf="@+id/imageBttn"** : verilen id nin **sağında** kendini konumlandırır.
- **android:layout_below="@+id/imageBttn"**: verilen id nin **altında** kendini konumlandırır.
- **android:layout_above="@+id/imageBttn"**: verilen id nin **yukarısında** kendini konumlandırır.

Dr.Öğ.Üyesi Ömer ÇETİN

# Dinamik Android RelativeLayout Oluşturma

```
<?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="match_parent">

    <Button
        android:id="@+id/bttn1"
        android:layout_toLeftOf="@+id/imageBttn"
        android:background="#FF4081"
        android:text="1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true" />

    <ImageButton
        android:id="@+id/imageBttn"
        android:src="@mipmap/ic_launcher"
        android:layout_centerInParent="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:id="@+id/bttn2"
        android:layout_toRightOf="@+id/imageBttn"
        android:background="#56afa7"
        android:text="3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true" />

    <Button
        android:id="@+id/bttn3"
        android:layout_above="@+id/imageBttn"
        android:background="#ecba88"
        android:text="2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true" />

    <Button
        android:id="@+id/bttn4"
        android:layout_below="@+id/imageBttn"
        android:background="#3F51B5"
        android:text="4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN


# RelativeLayout

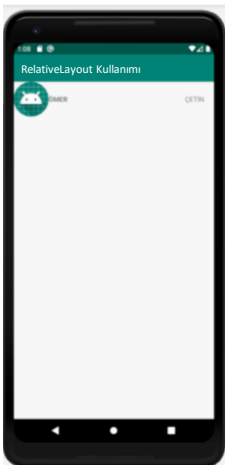
- Aşağıdaki XML layout yapısına baktığımızda; ImageView ve TextView'leri yatay olarak ortaladığımızı göreceksiniz.

  İlk TextView'in ImageView'in sağında yer alması için **android:layout_toRightOf="@+id/imageIcon"** kullandık.

**activity_main.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">

    <ImageView
        android:id="@+id/imageIcon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:src="@mipmap/ic_launcher" />

    <TextView
        android:id="@+id/txtSurname"
        android:layout_toRightOf="@+id/imageIcon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:text="ÖMER"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/place_distance"
        android:layout_marginRight="20dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:text="ÇETİN" />

</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN



# RelativeLayout

Dikkat etmemiz gereken :

- RelativeLayout da oluşturduğumuz **kuralları minimuma indirmeliyiz** ve Android bileşenleri arasında konumlandırmaları dikkatli bir şekilde yapmalıyız.
- Android Portrait ve Landscape olarak **iki farklı konuma göre** layout tasarımlarımızı oluşturmayı unutmayalım. Yani Android cihazınız dik durumdayken uygulama tasarımınız nasıl gözüküyorsa, ekranı yan çevirdiğimizde de o şekilde düzgün görünmesine dikkat edin. (İlerleyen derslerde ayrıntılı olarak anlatılacaktır.)
- Gereksiz yere **iç içe bir çok layout kullanmaktan kaçının**. Bu uygulamanızın performansını etkileyen faktörlerdendir. Daha kısa bir şekilde istediğiniz tasarımı oluşturmak varken bir çok karmaşık layout'ları iç içe kullanıp uygulamanızı gereksiz yere yormayın.



# Android Layout Türleri

1. LinearLayout
2. RelativeLayout
3. **FrameLayout**
4. GridLayout
5. TableLayout
6. ConstraintLayout

# Projeye Resim Eklemek

- Android Studio projenizi açın,
- Sol tarafta yer alan «res» klasörünü genişletin,
- «drawable» klasörüne sağ tıklayın,
- «Show in Explorer» seçeneğini seçin,
- Açılan gezinti penceresinde «drawable» klasörünü açın,
- Projeye eklemek istediğiniz dosyayı bu dizine kopyalayın ve istediğiniz ismi verin,
- Artık «@drawable/ dosya_adı» ile projenizden erişebilirsiniz.



# FrameLayout

FrameLayout ile Android **bileşenlerinin üst üste gelmesi**ni sağlayabiliriz. Örneğin bir resminiz var üzerinde isim yazmasını veya buton yer almasını istiyoruz işte o zaman FrameLayout kullanmamız en mantıklı çözüm olacaktır.


**activity_main.xml**

```
<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="center"
        android:src="@drawable/bmw" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:layout_gravity="center_horizontal|bottom"
        android:padding="10dp"
        android:background="#cccc"
        android:textColor="#ffffffff"
        android:text="Başla" />
</FrameLayout>
```

main_activity.java

```
package com.example.linearlayoutkullanm;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.graphics.Color;
import android.view.Gravity;
import android.widget.FrameLayout;
import android.widget.ImageView;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main);

        ImageView img = new ImageView(this);
        FrameLayout.LayoutParams imageParams = new
FrameLayout.LayoutParams(FrameLayout.LayoutParams.MATCH_PARENT,
                FrameLayout.LayoutParams.MATCH_PARENT);
        img.setImageResource(R.drawable.bmw);
        img.setLayoutParams(imageParams);

        TextView txt = new TextView(this);
        txt.setText("New York");
        txt.setTextColor(Color.parseColor("#ffffffff"));
        txt.setPadding(10,10,10,10);

        FrameLayout.LayoutParams txtParams =
                new FrameLayout.LayoutParams(FrameLayout.LayoutParams.WRAP_CONTENT,
                        FrameLayout.LayoutParams.WRAP_CONTENT);
        txtParams.gravity = Gravity.CENTER_HORIZONTAL | Gravity.BOTTOM;
        txt.setLayoutParams(txtParams);


        FrameLayout outsideLayout = new FrameLayout(this);
        outsideLayout.setLayoutParams(new
FrameLayout.LayoutParams(FrameLayout.LayoutParams.MATCH_PARENT,
FrameLayout.LayoutParams.MATCH_PARENT));
        outsideLayout.addView(img);
        outsideLayout.addView(txt);
        setContentView(outsideLayout);


    }
}
```

Dr.Öğ.Üyesi Ömer ÇETİN

## FrameLayout


# Dinamik Android FrameLayout Oluşturma

- Bir şarkı indirme programı düşünelim ekranımızın en sağında indirme butonu olsun bastığımızda progress çıksın ve indirmeye başlasın; indirme sırasında da indirmeden vazgeçtik diyelim ki iptal etme butonu olsun işte bu senaryo da uygulama ilk açıldığında indirme butonu aktif olarak gelecek indirme ikonuna bastığımızda aynı yerdeki indirme butonu kaybolacak yerine iptal etme butonu gelecek.
- Gözümüzde canlandırdıysak eğer işte bu durumda o butonların aynı yerde konumlanabilmesi için yine FrameLayout kullanmamız gerekir.
- ProgressBar ve TextView de indirme yüzdesi de yer almakta fakat o kısımları şimdilik unutun asıl mantık **indirme butonuna basıldığı anda iptal etme butonunun gösterilmesi ve ikisininde aynı yerde konumlandırma yapması**dır.

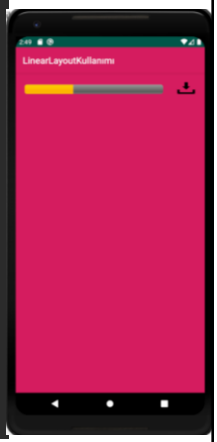
Dr.Öğ.Üyesi Ömer ÇETİN

# Dinamik Android FrameLayout Oluşturma

**activity_main.xml**

```
<RelativeLayout
    android:background="@color/colorAccent"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">


    <FrameLayout
        android:layout_marginRight="20dp"
        android:layout_marginTop="20dp"
        android:id="@+id/frameLayout"
        android:layout_alignParentEnd="true"
        android:layout_alignParentRight="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">


        <Button
            android:id="@+id/downloadButton"
            android:layout_width="40dp"
            android:layout_height="25dp"
            android:background="@drawable/download"
            android:paddingLeft="8dp"
            android:paddingRight="8dp"
            android:paddingTop="8dp" />

        <Button
            android:id="@+id/cancelButton"
            android:layout_width="40dp"
            android:layout_height="25dp"
            android:background="@drawable/ic_launcher_background"
            android:paddingLeft="8dp"
            android:paddingRight="8dp"
            android:paddingTop="8dp"
            android:visibility="invisible" />

    </FrameLayout>
```

```
<LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="1"
        android:id="@+id/linearLayout6">

    </LinearLayout>

    <ProgressBar
        android:id="@+id/progress"
        style="@android:style/Widget.ProgressBar.Horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/frameLayout"
        android:layout_marginRight="30dp"
        android:layout_marginLeft="30dp"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:max="100"
        android:progress="35"
        android:layout_toLeftOf="@+id/frameLayout"
        android:layout_toStartOf="@+id/frameLayout" />

    <TextView
        android:textColor="#cccc"
        android:layout_marginRight="5dp"
        android:gravity="center_vertical"
        android:text=""
        android:textSize="16sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtPercentage"
        android:layout_alignBottom="@+id/progress"
        android:layout_toLeftOf="@+id/frameLayout"
        />

</RelativeLayout>
```






# Dinamik Android FrameLayout Oluşturma

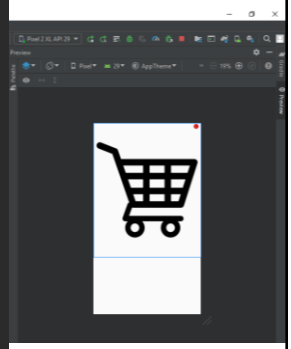
- **Notification Badge:** Bildirim veya mesaj geldiğinde o bildiriminin sayısının kaç olduğunu göstermek için çoğunlukla bu şekilde bir gösterim tercih ederiz.
- Bu yapı için hazır third party kütüphaneler var tabiki de fakat FrameLayout kullanımını daha iyi gözümüzde canlandırmak için XML layout yapısını oluşturduk.

# Dinamik Android FrameLayout Oluşturma

- ImageView (alışveriş sepeti simgesi için) ve TextView ile (sayı değeri için) özel bir düzen oluşturun…

layout/custom_action_item_layout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    style="?attr/actionButtonStyle"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:clipToPadding="false"
    android:focusable="true">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="right"
        android:src="@drawable/ic_action_cart"/>

    <TextView
        android:id="@+id/cart_badge"
        android:layout_width="20dp"
        android:layout_height="20dp"
        android:layout_gravity="right|end|top"
        android:layout_marginEnd="-5dp"
        android:layout_marginRight="-5dp"
        android:layout_marginTop="3dp"
        android:background="@drawable/badge_background"
        android:gravity="center"
        android:padding="3dp"
        android:textColor="@android:color/white"
        android:text="0"
        android:textSize="10sp"/>
</FrameLayout>
```



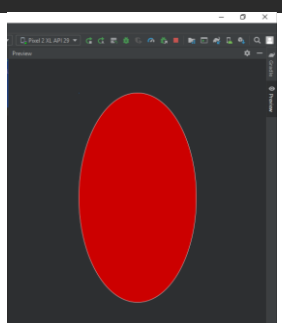


# Dinamik Android FrameLayout Oluşturma

- Şekil kullanarak çizilebilir dairesel rozet arka planı oluşturun.

drawable/badge_background.xml:

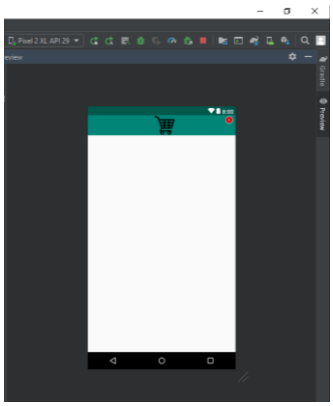
```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android" android:shape="oval">

    <solid android:color="@android:color/holo_red_dark"/>
    <stroke android:color="@android:color/white" android:width="1dp"/>

</shape>
```



# Dinamik Android FrameLayout Oluşturma

- Menü öğesine özel düzen ekleyin.
  menu/main_menu.xml

```
<menu
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto" >

    <item
        android:id="@+id/action_cart"
        android:icon="@drawable/ic_action_cart"
        android:title="Cart"
        app:actionLayout="@layout/custom_action_item_layout"
        app:showAsAction="always"/>

</menu>
```





# Dinamik Android FrameLayout Oluşturma

- MainActivity'de, aşağıdaki kodları ekleyelim:

```
package com.example.badge_application;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.view.MenuItemCompat;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    TextView textCartItemCount;
    int mCartItemCount = 10;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);

        final MenuItem menuItem = menu.findItem(R.id.action_cart);

        View actionView = MenuItemCompat.getActionView(menuItem);
        textCartItemCount = (TextView) actionView.findViewById(R.id.cart_badge);

        setupBadge();

        actionView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                onOptionsItemSelected(menuItem);
            }
        });

        return true;
    }
```

```
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {

        switch (item.getItemId()) {

            case R.id.action_cart: {
                // Do something
                return true;
            }
        }
        return super.onOptionsItemSelected(item);
    }

    private void setupBadge() {

        if (textCartItemCount != null) {
            if (mCartItemCount == 0) {
                if (textCartItemCount.getVisibility() != View.GONE) {
                    textCartItemCount.setVisibility(View.GONE);
                }
            } else {
                textCartItemCount.setText(String.valueOf(Math.min(mCartItemCount,
99)));
                if (textCartItemCount.getVisibility() != View.VISIBLE) {
                    textCartItemCount.setVisibility(View.VISIBLE);
                }
            }
        }
    }

}
```

# Dinamik Android FrameLayout Oluşturma



# Android Layout Türleri

1. LinearLayout
2. RelativeLayout
3. FrameLayout
4. **GridLayout**
5. TableLayout
6. ConstraintLayout

# GridLayout

- GridLayout içinde yer alan Android bileşenlerini bir dikdörtgen alana yerleştiren bir layout türüdür. İçindeki alanı hücrelere ayırdığı gibi satır ve sütün sayılarını da kendimiz belirleyebiliriz.
- Bu layout türünü daha iyi anlayabilmek için hesap makinesinin XML layout kısmını tasarlayacağız.
- GridLayout ağırlık olarak tanımlanan bileşenler için ağırlık prensibini desteklemez. Örneğin dört butonu eşit boyutlarda aynı satır da göstermeyi planladığımızda butonlar bazı cihazlarda eşit bir şekilde konumlanmış gözükürken, bazı cihazlarda butonlarda ekrana sığmama dolayısıyla eşit ölçüde konumlandırmama durumu olabilir.
- İşte bu durumda GridLayout yerine Api 21 ile gelen (android.support.v7.widget.GridLayout) kullanılabilir.



# GridLayout

- GridLayout için xml de kullanacağımız önemli özelliklerin neler olduğuna bakalım:
  - **android:columnCount**: Layout un içinde yer alan android bileşenlerinin maksimum sütun sayısını belirleyip, bu android bileşenlerin otomatik olarak konumlandırılmasını sağlar.
  - **android:rowCount**: Layout un içinde yer alan android bileşenlerinin maksimum satır sayısını belirleyip, bu android bileşenlerin otomatik olarak konumlandırılmasını sağlar.
  - **android:useDefaultMargins**: true olarak tanımlandığında varsayılan kenar boşlukları kullanmasını sağlar.
  - **android:orientation**: Layoutu yatay (horizontal) veya dikey (vertical) biçimde konumlandırmamızı sağlar.

# GridLayout

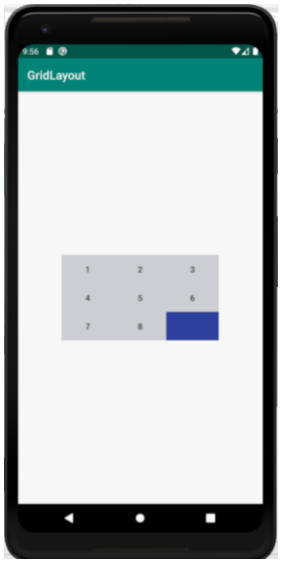
- Aşağıdaki kod yapısında columnCount=3 yaparak satır ve sütun sayılarını belirledik. GridLayout içine yerleştirdiğimiz Butonlarda aşağıdaki ekran çıktısında göreceğiniz gibi kendini otomatik olarak konumlandırdı. GridLayout arka plan rengini mavi yaptık bu yüzden de boşta kalan kısımda mavi gözüküyor eğer bir buton daha olsaydı oraya konumlanacaktı.

**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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <GridLayout
            android:background="#303F9F"
            android:layout_centerHorizontal="true"
            android:layout_centerInParent="true"
            android:columnCount="3"
            android:rowCount="3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="1" />

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="2" />
            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="3" />
            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="4" />

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="5" />

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="6" />

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="7" />

            <Button
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:background="#CFD0D3"
                android:text="8" />

        </GridLayout>

    </RelativeLayout>

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



# GridLayout

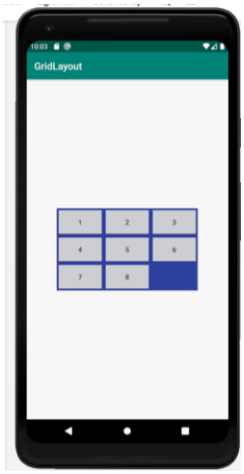


# GridLayout

- **useDefaultMargins true** olarak ayarladığımızda varsayılan kenar boşluklarını ayarlar ve aşağıdaki ekran çıktısında gördüğünüz gibi butonlar arasında boşluklar oluşturulur.

**activity_main.xml**

```
<GridLayout
    android:background="#303F9F"
    android:layout_centerHorizontal="true"
    android:layout_centerInParent="true"
    android:useDefaultMargins="true"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:columnCount="3"
    android:rowCount="3"
    android:orientation="horizontal">
```



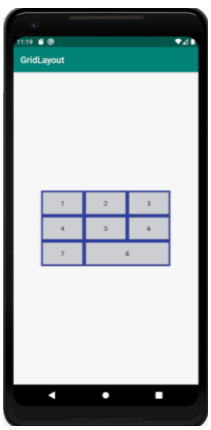



# GridLayout

- Yukarıda ki kod yapısında gözlemlediğimiz gibi bir satır alan boş kalmıştı. Şimdi o alanı 8 numaralı buton ile doldurmaya çalışmak istersek neler yapacağımıza bakalım; 8 numaralı butona **android:layout_columnSpan=”2”** diyerek aslında 2 birim kolun yer almasını ve **android:layout_gravity=”fill”** ile de o kapladığı alan kadar olan yeri doldurmasını sağladık.

**activity_main.xml**

```
<Button
    android:layout_columnSpan="2"
    android:layout_gravity="fill"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:background="#CFD0D3"
    android:text="8" />
```



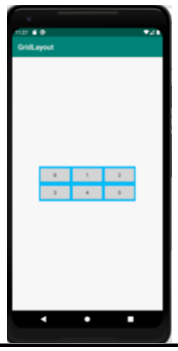
# Dinamik Android GridLayout Oluşturma

**MainActivity.java**

```
package com.example.gridlayout;

import androidx.appcompat.app.AppCompatActivity;

import android.graphics.Color;
import android.os.Bundle;
import android.view.Gravity;
import android.widget.Button;
import android.widget.GridLayout;
import android.widget.RelativeLayout;

public class MainActivity extends AppCompatActivity {
        Button bttn;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            //setContentView(R.layout.activity_main);

            GridLayout gridLayout = new GridLayout(MainActivity.this);
            gridLayout.setBackgroundColor(Color.parseColor("#00bfff"));
            gridLayout.setColumnCount(3);
            gridLayout.setRowCount(3);
            for (int i = 0; i<6; i++) {
                bttn = new Button(MainActivity.this);
                bttn.setText(""+i);
                GridLayout.LayoutParams param =new GridLayout.LayoutParams();
                param.height = GridLayout.LayoutParams.WRAP_CONTENT;
                param.width = GridLayout.LayoutParams.WRAP_CONTENT;
                bttn.setLayoutParams(param);
                gridLayout.addView(bttn, i);
            }


            RelativeLayout.LayoutParams layoutparams = new RelativeLayout.LayoutParams(
                    RelativeLayout.LayoutParams.MATCH_PARENT,
                    RelativeLayout.LayoutParams.MATCH_PARENT
            );

            RelativeLayout outsideLayout = new RelativeLayout(MainActivity.this);
            outsideLayout.setLayoutParams(layoutparams) ;
            outsideLayout.addView(gridLayout);
            outsideLayout.setGravity(Gravity.CENTER);
            setContentView(outsideLayout);

        }
    }
```

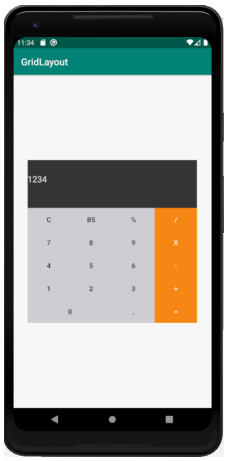
Dr.Öğ.Üyesi Ömer ÇETİN

• Yandaki kodlamaya baktığımızda dinamik bir şekilde GridLayout oluşturduk, setBackGroundColor ile arka plan rengini belirledik, setColumnCount ve setRowCount ile sütun ve satır sayılarını belirledik. Toplamda 6 adet butonu GridLayout içine yerleştirdik. Daha sonra RelativeLayout oluşturduk en dışta bu layout olacağından RelativeLayout içine de GridLayout konumlandırmasını yaptık.




# GridLayout

• Şimdiye kadar öğrendiğimiz modüller ile bir hesap makinesinin tasarımını yapalım; aşağıdaki kodlamaya baktığımızda GridLayout için sütun sayısını 4 olarak belirledik. Hemen içinde yer alan EditText e **android:layout_columnSpan=”4”** dedik çünkü yanında başka bir android bileşeni olmasını istemiyoruz o alanda sadece EditText ile oluşmasını istediğimiz için columnSpan kullanarak GridLayout da belirlediğimiz sütun sayısını burada da kullandık. Text değeri 0 olan buton içinde android:layout_columnSpan=”2” ve android:layout_gravity=”fill” değerlerini set ederek 2 birimlik yer kaplamasını sağladık.




Dr.Öğ.Üyesi Ömer ÇETİN

```
<RelativeLayout
    android:id="@+id/relative"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">


    <GridLayout
        android:layout_centerHorizontal="true"
        android:layout_centerInParent="true"
        android:columnCount="4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

        <EditText
            android:layout_columnSpan="4"
            android:textColor="#fff"
            android:background="#363636"
            android:layout_width="match_parent"
            android:layout_height="100dp"
            />


        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="#CFD0D3"
            android:text="C" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="BS" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="%" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#FA8910"
            android:textColor="#FFFF"
            android:text="/" />
```

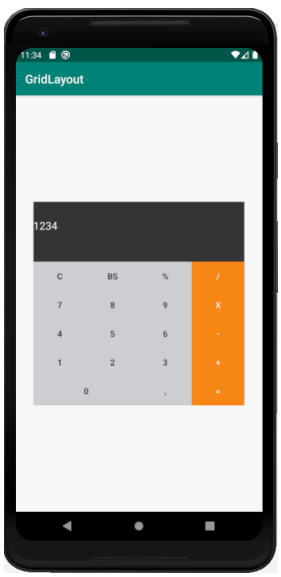


```
        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="7" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="8" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="9" />

        <Button
            android:textColor="#FFFF"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#FA8910"
            android:text="x" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="4" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="5" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="6" />

        <Button
            android:textColor="#FFFF"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#FA8910"
            android:text="-" />
```

```
        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="1" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="2" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="3" />

        <Button
            android:textColor="#FFFF"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#FA8910"
            android:text="+" />

        <Button
            android:layout_columnSpan="2"
            android:layout_gravity="fill"
            android:background="#CFD0D3"
            android:text="0" />


        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="," />

        <Button
            android:textColor="#FFFF"
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#FA8910"
            android:text="=" />


    </GridLayout>
```



# GridLayout



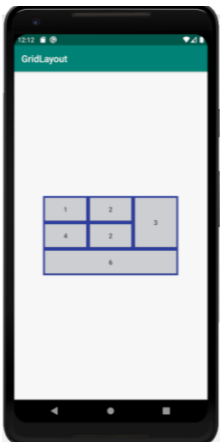
# GridLayout

- Aşağıdaki kodlamaya baktığımızda ise; text değeri 3 olan butonun android:layout_rowSpan=”2” ve android:layout_gravity=”fill_vertical” değerlerini vererek dikey olarak 2 birimlik yer kapladık. text değeri 6 olan butonunun kullanımınıda benzerini de ayrıntılı bir şekilde anlattık.

activity_main.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="match_parent">


    <GridLayout
        android:background="#303F9F"
        android:layout_centerHorizontal="true"
        android:layout_centerInParent="true"
        android:useDefaultMargins="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:columnCount="3"
        android:rowCount="3"
        android:orientation="horizontal">

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="1" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="2" />
```

Dr.Öğ.Üyesi Ömer ÇETİN




# GridLayout

- GridLayout orientation değerini dikey ( vertical ) olarak belirttiğimizde ise aşağıda gördüğünüz gibi bir ekran görüntüsünü elde ettik.

activity_main.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="match_parent">


    <GridLayout
        android:background="#303F9F"
        android:layout_centerHorizontal="true"
        android:layout_centerInParent="true"
        android:useDefaultMargins="true"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:columnCount="3"
        android:rowCount="3"
        android:orientation="vertical">

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="1" />

        <Button
            android:layout_height="wrap_content"
            android:layout_width="wrap_content"
            android:background="#CFD0D3"
            android:text="2" />

        …
</RelativeLayout>
```

Dr.Öğ.Üyesi Ömer ÇETİN