آموزش برنامه نویسی اندروید
در محیط اندروید استودیو

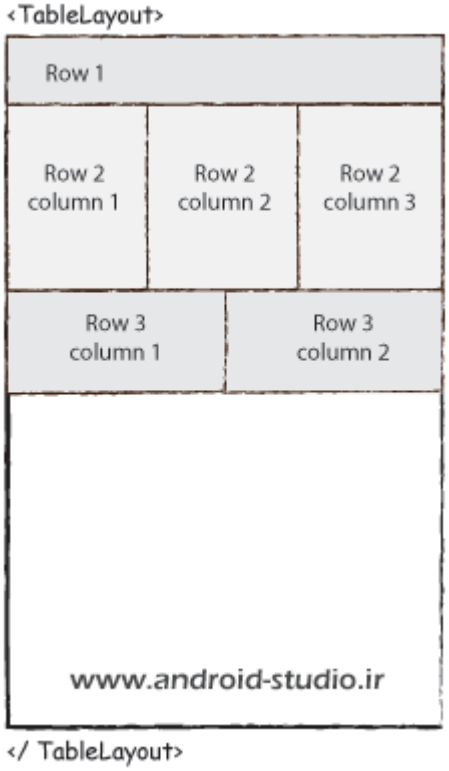
TableLayout

مدرس : سید مهدی مطهری

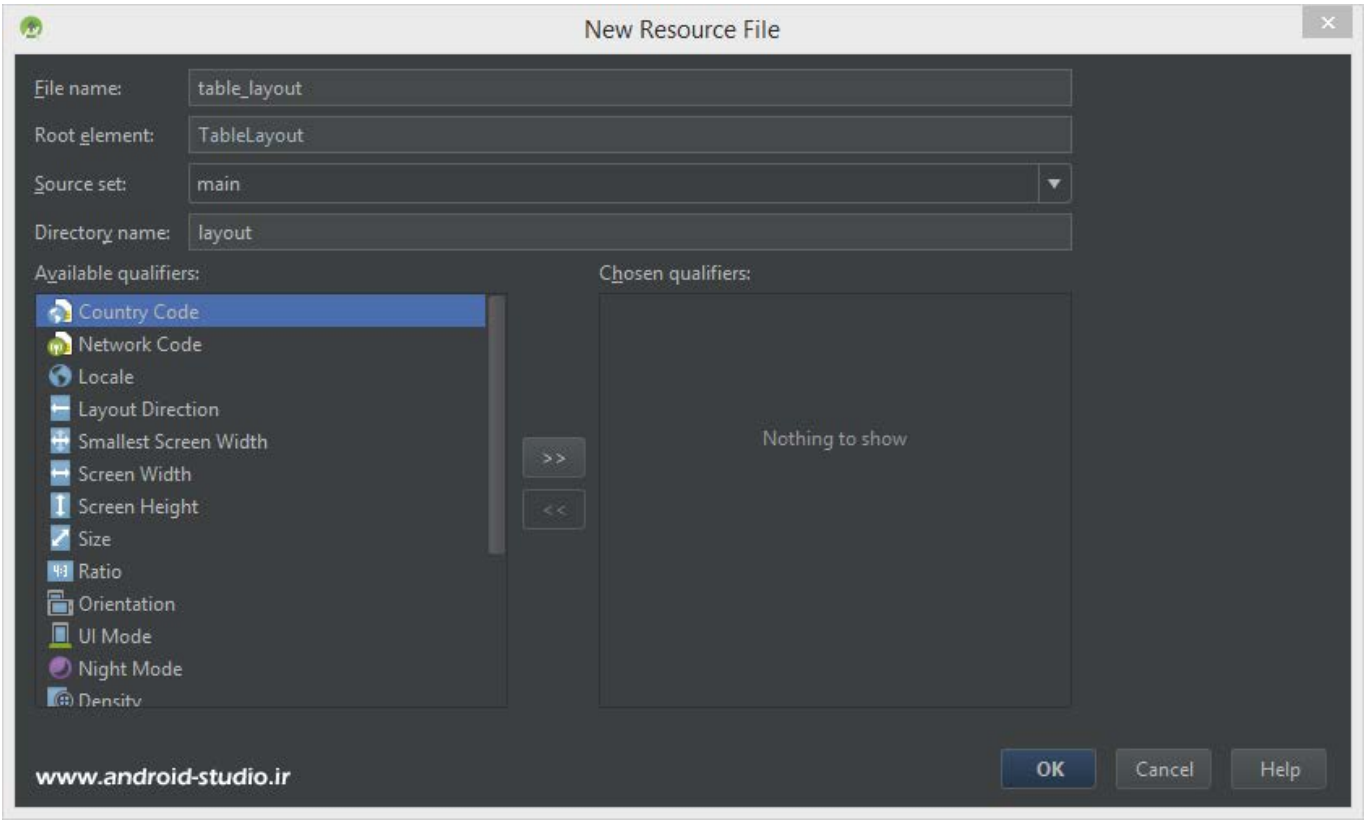
www.android-studio.ir

به نام خدا

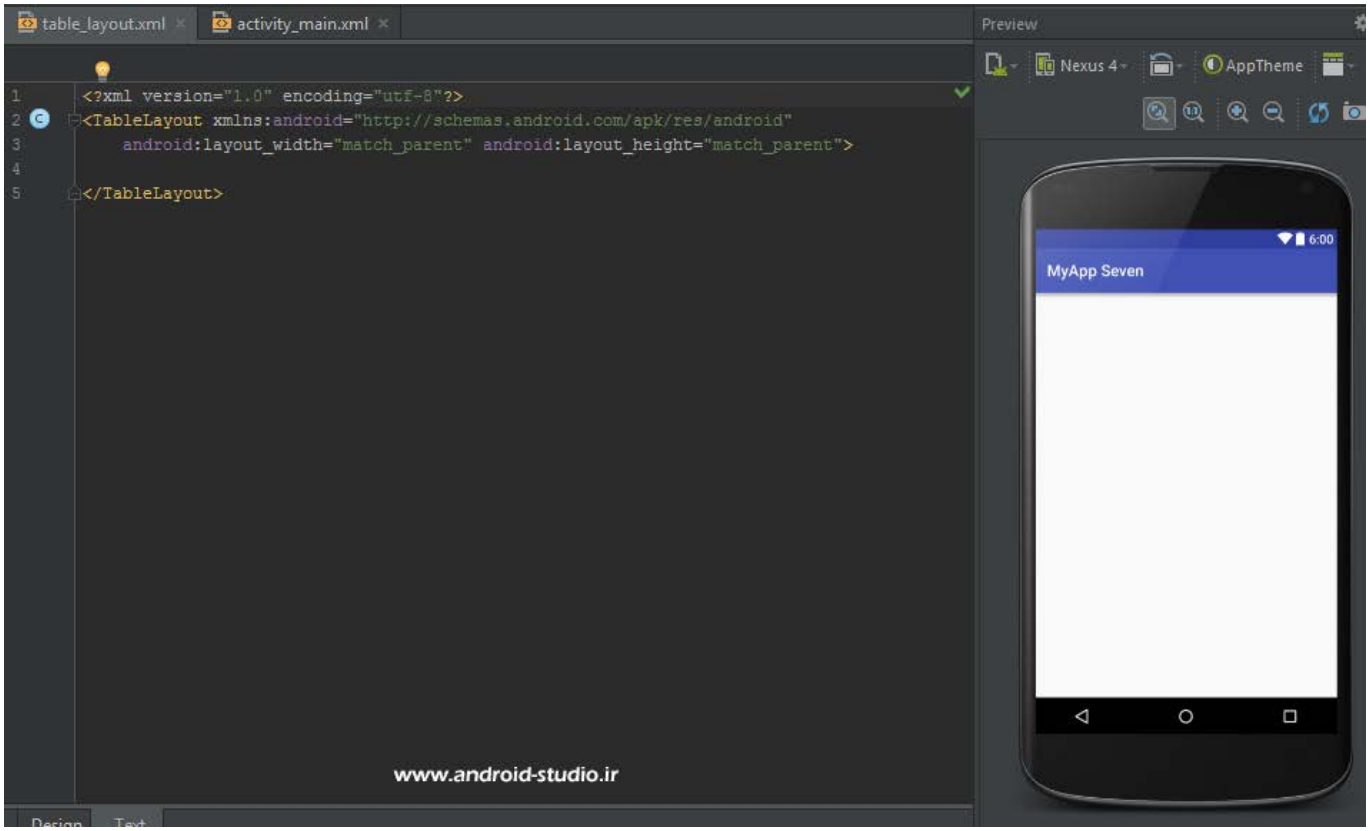
در آموزش طراحی رابط کاربری به معرفی دو نوع ViewGroup با نام های RelativeLayout و LinearLayout پرداختیم. در این مقاله TableLayout را بررسی می کنیم.



اگر با زبان طراحی صفحات وب Html آشنایی داشته باشید، مبحثی با عنوان Table وجود دارد که عناصر مدنظر طراح را به صورت افقی و عمودی در سطرها و ستون های مختلف در کنار یکدیگر قرار می دهد. در اندروید نیز مشابه جدول بندی در صفحات وب عمل می کنیم.مطابق آنچه قبلا توضیح دادیم، اندروید استودیو را باز نموده و در پوشه layout یک TableLayout با نام دلخواه می سازیم.



که پس از تایید، تگ TableLayout با دو ویژگی layout_width و layout_height ایجاد می شود. اینکه اندروید استودیو به صورت پیش فرض این دو ویژگی را اضافه می کند نشان از اهمیت وجود این موارد می باشد. همیشه باید عرض و ارتفاع لایه مشخص باشد.



جدول از سطر (Row) و ستون (Column) تشکیل شده است. در اندروید سطرها توسط تگ باز و بسته <TableRow></TableRow>
تعریف می شوند. ستون ها داخل سطرها قرار گرفته و توسط خود اندروید به ترتیب در کنار یکدیگر قرار می گیرند. برای مثال می خواهیم سه Button را در یک سطر در کنار یکدیگر به صورت افقی قرار دهیم. یک TableRow ساخته و عرض و ارتفاع را تعیین می کنیم :

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

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

    </TableRow>

</TableLayout>
```

سپس سه دکمه داخل TableRow تعریف می کنیم :

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

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content">


        <Button
            android:text="Button One"
            />

        <Button
            android:text="Button Two"
            />

        <Button
            android:text="Button Three"
            />

    </TableRow>

</TableLayout>
```

مشاهده می کنیم که سه دکمه در سه ستون و به صورت افقی در کنار یکدیگر قرار گرفته اند :

بر خلاف Html در اندروید نیاز به تعریف ستون ها نیست بلکه هر عنصر به عنوان یک ستون درنظر گرفته می شود.حالا ما سه ستون داریم. ستون ها به ترتیب قرار گیری دارای شماره هستند که مقداردهی از عدد صفر شروع می شود. در مثال بالا، دکمه اول دارای مقدار ۰ ، دکمه دوم مقدار ۱ و دکمه سوم مقدار ۲ را به خود اختصاص داده اند.این مقداردهی کاربردهایی دارد.فرض کنیم می خواهیم در سطر بعدی، دو دکمه داشته باشیم که دکمه اول به اندازه یک ستون از ابتدای صفحه فاصله داشته باشد. این خاصیت با ویژگی layout_column تعریف می شود. وقتی برای دکمه اول، مقدار layout_column را برابر با عدد ۱ قرار دهیم به این معنی است که باید در ستون دوم قرار گیرد :

```
<!-- Column One -->
<TableRow
android:layout_width="match_parent"
android:layout_height="wrap_content">

<Button
    android:text="Button One"
    />

<Button
    android:text="Button Two"
    />

<Button
    android:text="Button Three"
    />

</TableRow>

<!-- Column Two -->
```

```
<TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
        android:text="Button One"
        android:layout_column="1"
        />

    <Button
        android:text="Button Two"
        />

</TableRow>
```

دکمه Button One سطر دوم در زیر Button Two سطر اول قرار گرفته و اگر مجدد مقدار ۲ را قرار دهیم، در زیر دکمه Button Three سطر اول قرار خواهد گرفت.

ویژگی دیگری داریم تحت عنوان layout_span که به واسطه آن میزان پوشش یک عنصر را به نسبت ستون ها مشخص می کند. می خواهیم در مثال بالا، به جای آنکه دکمه اول ستون دوم از ابتدای صفحه به اندازه یک ستون فاصله بگیرد، به اندازه دو ستون فضا اشغال کند. بنابراین مقدار ۲ را برای این ویژگی درنظر می گیریم (این عدد تعداد ستون را تعیین می کند و با شماره ستون ها متفاوت است).

```
<!-- Column One -->
<TableRow
android:layout_width="match_parent"
android:layout_height="wrap_content">

<Button
    android:text="Button One"
    />

<Button
    android:text="Button Two"
    />

<Button
    android:text="Button Three"
    />

</TableRow>

<!-- Column Two -->
<TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
        android:text="Button One"
        android:layout_span="2"
        />

    <Button
        android:text="Button Two"
        />

</TableRow>
```

اگر دقت کنید در انتهای سمت راست صفحه، فضای خالی مشاهده می شود.علت این است که سه دکمه ما نتوانسته تمام عرض صفحه را پوشش دهد. برای اینکه عناصر تمام عرض صفحه را پوشش دهند، از ویژگی stretchColumns در TableLayout استفاده می کنیم. اگر مقدار این ویژگی برابر "*" قرار گیرد، تمامی ستون ها به یک اندازه کشیده می شوند تا فضای خالی از بین برود و اگر مقدار را برابر شماره هر یک از ستون ها قرار دهیم، فقط همان ستون به نحوی از دو طرف کشیده می شود تا فضای خالی از بین برود :

حالت اول :

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

    <!-- Column One -->
    <TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
        android:text="Button One"
        />

    <Button
        android:text="Button Two"
        />

    <Button
```

```
        android:text="Button Three"
        />

    </TableRow>

    <!-- Column Two -->
    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:text="Button One"
            android:layout_span="2"
            />

        <Button
            android:text="Button Two"
            />

    </TableRow>


</TableLayout>
```

خروجی :

مشاهده می کنیم همه عناصر به نحوه فضا را اشغال کردند که فضای خالی باقی نماند.

در حالت دوم مقدار صفر یعنی ستون اول را به stretchColumns می دهیم :

```
android:stretchColumns="0"
```

فقط ستون اول (Button One) فضای خالی را از بین برد و اندازه سایر ستون ها دست نخورده باقی ماند. اگر هم بخواهیم این ویژگی به عنوان مثال برای ستون های اول و سوم اعمال شود، مقدار دو ستون را به این صورت در ویژگی قرار می دهیم :

```
android:stretchColumns="0,2">
```

توسط ویژگی collapseColumns نیز می توان ستون یا ستون هایی را از دید کاربر مخفی نمود :

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:stretchColumns="0,2"
    android:collapseColumns="0">

    <!-- Column One -->
    <TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
        android:text="Button One"
        />

    <Button
        android:text="Button Two"
        />

    <Button
        android:text="Button Three"
        />

    </TableRow>

    <!-- Column Two -->
    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:text="Button One"
            android:layout_span="2"
            />

        <Button
            android:text="Button Two"
            />

    </TableRow>


</TableLayout>
```

با قرار دادن مقدار صفر برای این ویژگی، ستون یک که شامل دکمه های Button One بود از دید کاربر حذف گردید.