
بسم الله الرحمن الرحيم

عنوان کتاب : Head First Android Development

مترجم : حسن زمانی

استفاده از این کتاب با ذکر سه صلوات رایگان هست :

اولی برای شفای همه مریض ها

دومی برای خوب شدن همه بیماران جسمی

و سومی هم برای شفای همه بیماران روحی

WWW.HZAVR.IR

HEADFIRST@HZAVR.IR

A Brain-Friendly Guide

Head First Android Development

فصل دوم

Learn what matters, when it matters

Load important concepts directly into your brain

Bend your mind around dozens of puzzles and exercises

2

Avoid embarrassing mistakes

Master out-of-this-world concepts

O'REILLY®

Jonathan Simon


ბედნიერი ახალი წელი
سال نو مبارک

مـ قدمه:

باز هم بهار شد و شکوفه ها شکفتند و سبزها سر سبز شدند و بلبل ها چه چه میزنند و نـیم

خنک بهاری می وزد و در کنار همه این حرفهای قشنگ دانشگاهها هم تعطیل شدند و فرصتی شد تا چند صفحه دیگر از کتاب زیبای هد فرست اندروید را ترجمه کنم و در اختیار همه شما علاقه مندان برنامه نویـی اندروید قرار بدم . گرچه قرار بود بیشتر از یک فصل ترجمه کنم و البته این کار هم شده اما به صورت خام هـت و به طور کامل اماده نشده , بنابراین فعلا ترجمه فصل دوم را منتشر میکنم و در اینده ای نچندان دور (چشم به هم بزنید رسیده) فصلهای اینده نیز منتشر میشود .

سال خوبی داشته باشید .

حسن زمانی

بهار 1394

2 به برنامه‌ات فعالیت اضافه کن

اضافه کردن رفتار و فعالیت

برنامه ها تعاملی هستند ! وقتی به برنامه خودتون رفتار اضافه کنید این رفتار باعث میشه که کاربرها بتونن با اون کارهایی را انجام بدن و این همون چیزیه که باعث می شه کاربرها اون را دوست داشته باشن .

همون طور که در بخش اول دیدید اندروید واقعا ظاهر و بخش بصری برنامه را از کدهای جاوا جدا می کنه (لایه های XML و منبع رشته ها را به خاطر بیارین !) . در این بخش شما تعدادی رفتار و فعالیت به برنامه Android love خودتون اضافه می کنید. و در طول این کار شما یاد می گیرید که چطور فایلهای XML و جاوا بدون هیچ نقصی با هم کار میکنن و به شما قدرت خلق برنامه های اندروید را میدن .

برنامه خودتون را تعاملی کنید

بله ما می خواهیم که اون کاری
انجام بده ! فکر من اینه که شعرها
را مخفی کنیم و یک دکمه به برنامه
اضافه کنیم و وقتی طرفدارانمون
اون را فشار میدن شعرها هم ظاهر
بشن ! بزار برات بکشم....

ما نمی خواهیم برنامه ما خشک و ساکن
باشه ! ولی این طور که میبینیم اون
فقط شعرهای haiku را نشون میده ...

بزار ببینیم Pajama Death چی توی سرشون دارن .....

برنامه Pajama Death که به اون یک دکمه اضافه شده

Pajama Death برات برنامه ای که توی سرشون هست را کشیدن بنابراین تو میتونی اون را بسازی.

اونها یک دکمه به بالا اضافه کردن ، و شعرها موقعی که برنامه اجرا می شه مخفی هستند . سپس وقتی دکمه فشار داده می شه شعرها نشون داده می شن !

اگه فکر می کنه این خوبه اما نمیدونی از کجا شروع کنی .... پس صفحه را ورق بزن !

Do this!

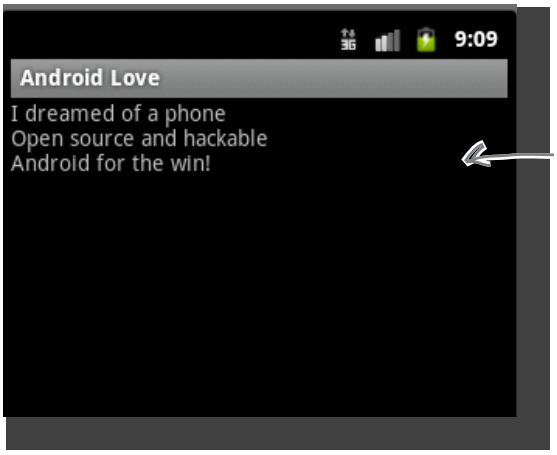
Open the AndroidLove project from Chapter 1 if you don't already have it open.

این جا بهت نشون داده شده که اون را چطور انجام بدی

چندتا کار هست که باید انجام بدی . پس اون را به قطعات کوچکتری تقسیم کن .

اولش ، پروژه برنامه را اجرا کن و اصلاحاتی روی اون انجام بده .

اگه پروژه Android Love را بستی ، اون را باز کن .



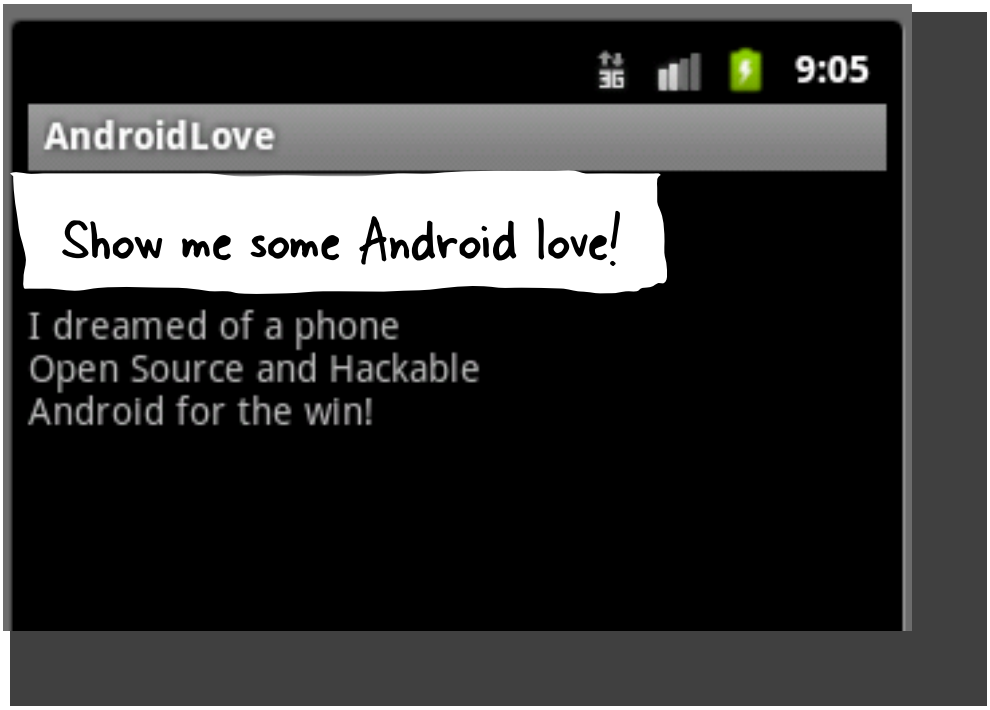
The AndroidLove app as you left it at the end of the last chapter.

1 دکمه به اون اضافه کن

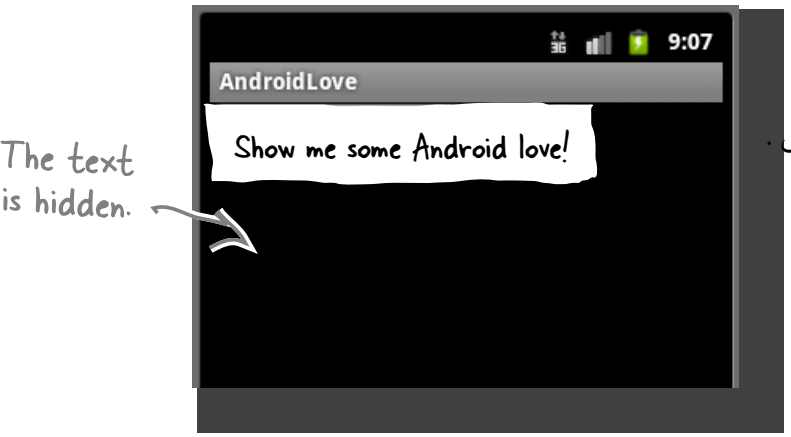
ما قصد داریم یک دکمه جدید به برنامه ات اضافه کنیم و در اخر این دکمه شعرها را به شما نشون میده.

اما برای شروع تو یاد می گیری که چطور یک قطعه جدید به صفحه ات اضافه کنی و می بینی که اصلا چه قطعات دیگه ای هستند و چطور باید اونها را به صفحه خودت اضافه کنی.

The new button.



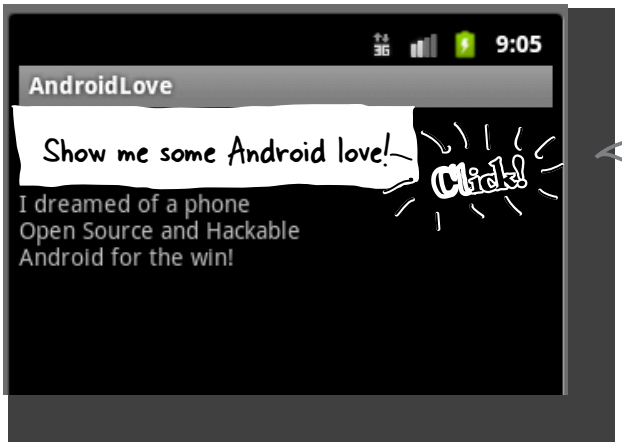
2 متن شعر را مخفی کن



بعد از این که دکمه را اضافه کردی ، تو باید متن شعرها را مخفی کنی . هنوز دکمه کاری انجام نمیده اما یک کم پیشرفت کردی ! تو متن شعرها را مخفی کردی . اینجاست که تو یاد می گیری که چطور با خصوصیتهای مختلف ویجتهای روی صفحه (مترجم: منظورش متن ها ، دکمه ها ، چک باکسها و ...هست) از طریق فایلهای XML کار کنی.

3 کاری کن که دکمه شعرها را نشون بده

بعدش باید کاری کنی که دکمه تو شعرها را نشون بده . در اینجاست که شما طعم کد نویس با جاوا را می چشید و این کدها را به فایلهای XML وصل می کنید . و اینجاست که جادویی اتفاق میفته !



BRAIN POWER

حالا که پروژه را باز کردید و اماده انجام کار هستید در اولین گام باید که دکمه را اضافه کنید .

به نظر شما کدام فایل را باید باز کنید تا بتوانید دکمه را اضافه کنید ؟

دکمه را اضافه کن

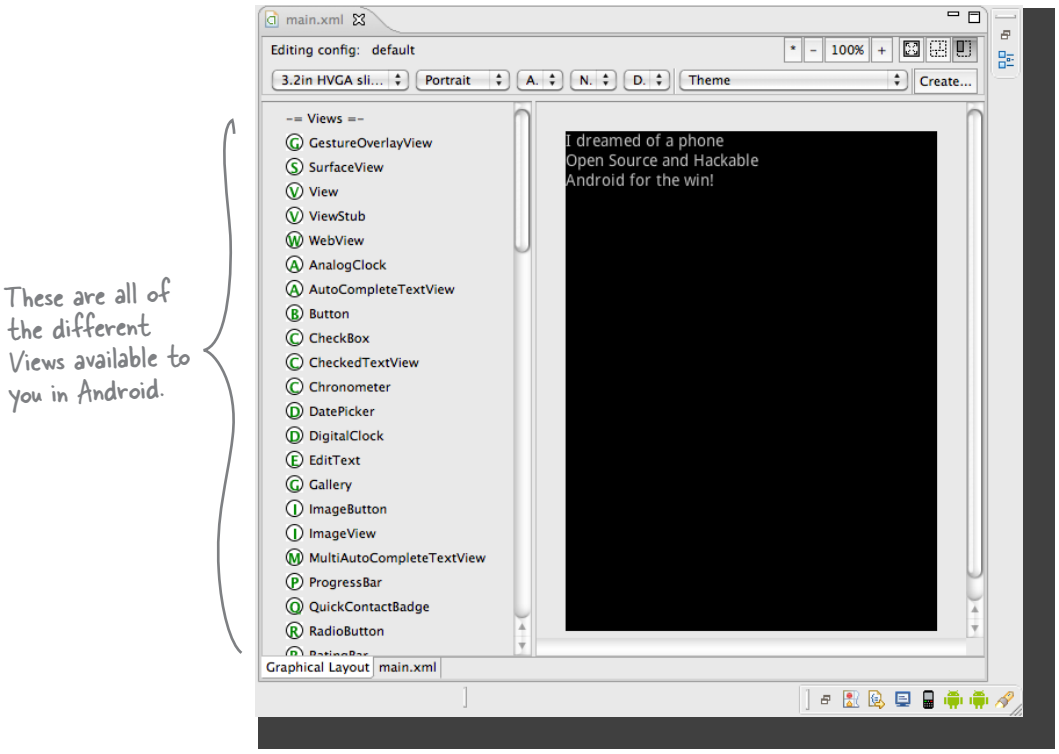
تو با main.xml در بخش اول کار کردی و میدونی که این فایل لایه کلی برای صفحه برنامه ات را مشخص می کنه . اینجا جاییه که تو باید دکمه جدید را به برنامه اب اضافه کنی . فایل main.xml ر ا باز کن .این فایل را در مسیر زیر میتونی پیدا کنی :

/res/layout/main.xml

در بخش 1 تو این لایه را از طریق کدهای XML ویرایش کردی . حالا با هم میریم تا دکمه را با استفاده از ویرایشگر گرافیکی به صفحه اضافه کنیم .

بر روی تب Graphical Layout کلیک کن تا لایه را به صورت گرافیکی مشاهده کنی .

حالا میبینی که همه Veiw ها در سمت چپ لیست شده اند .



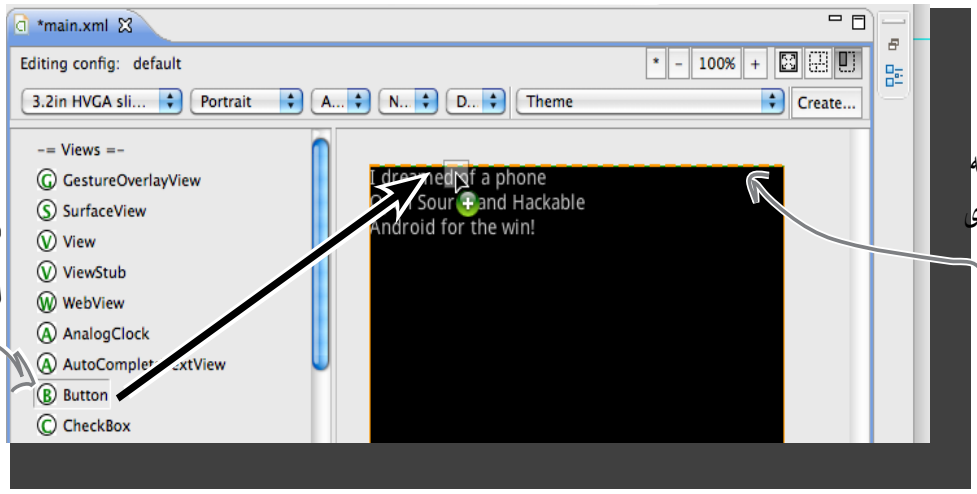
تو میتونی View ها را با استفاده از درگ کردن اونها روی صفحه منتقل کنی .

Adding a View Up Close

اضافه کردن View به بالای برنامه

اجازه بدید از نزدیک اضافه کردن دکمه را بوسیله ویرایشگر گرافیکی برسی کنیم .

1 روی button در سمت چپ کلیک کن و اون را به بالای لایه گرافیکی بکش . در اینجا متوجه میشی که خط چین هایی در جایی که قراره دکمه قرار بگیره ظاهر میشه . مطمعن شو که این خط چینها در بالا قرار میگیرن.



روی button کلیک کن

اون را به بالای لایه درگ کن

دکمه را به بالای لایه درگ کن و میبینی که

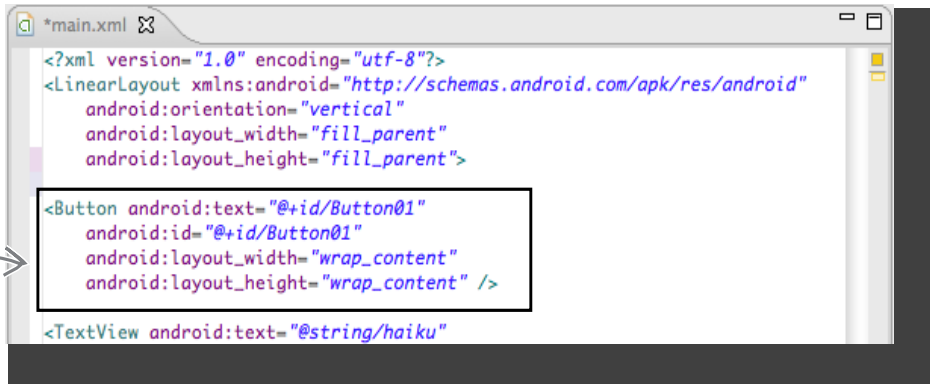
جایی که قراره دکمه قرار بگیره خط چینهای

نارنجی ظاهر میشن

این دکمه ای هست که تو اضافه کردی

بعد از این که دکمه را اضافه کردی چنین چیزی میبینی .

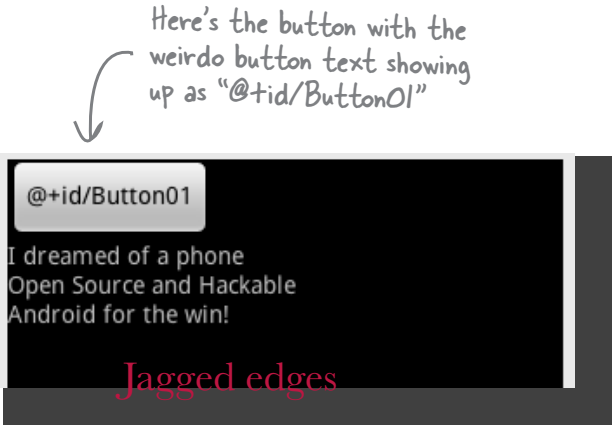
2 حالا برگرد به فایل main.xml تا کدهای XML اون را ببینی . اونجا اولین View خودت یعنی Button را که اضافه کردی می بینی !



The added XML declaration to create the Button.

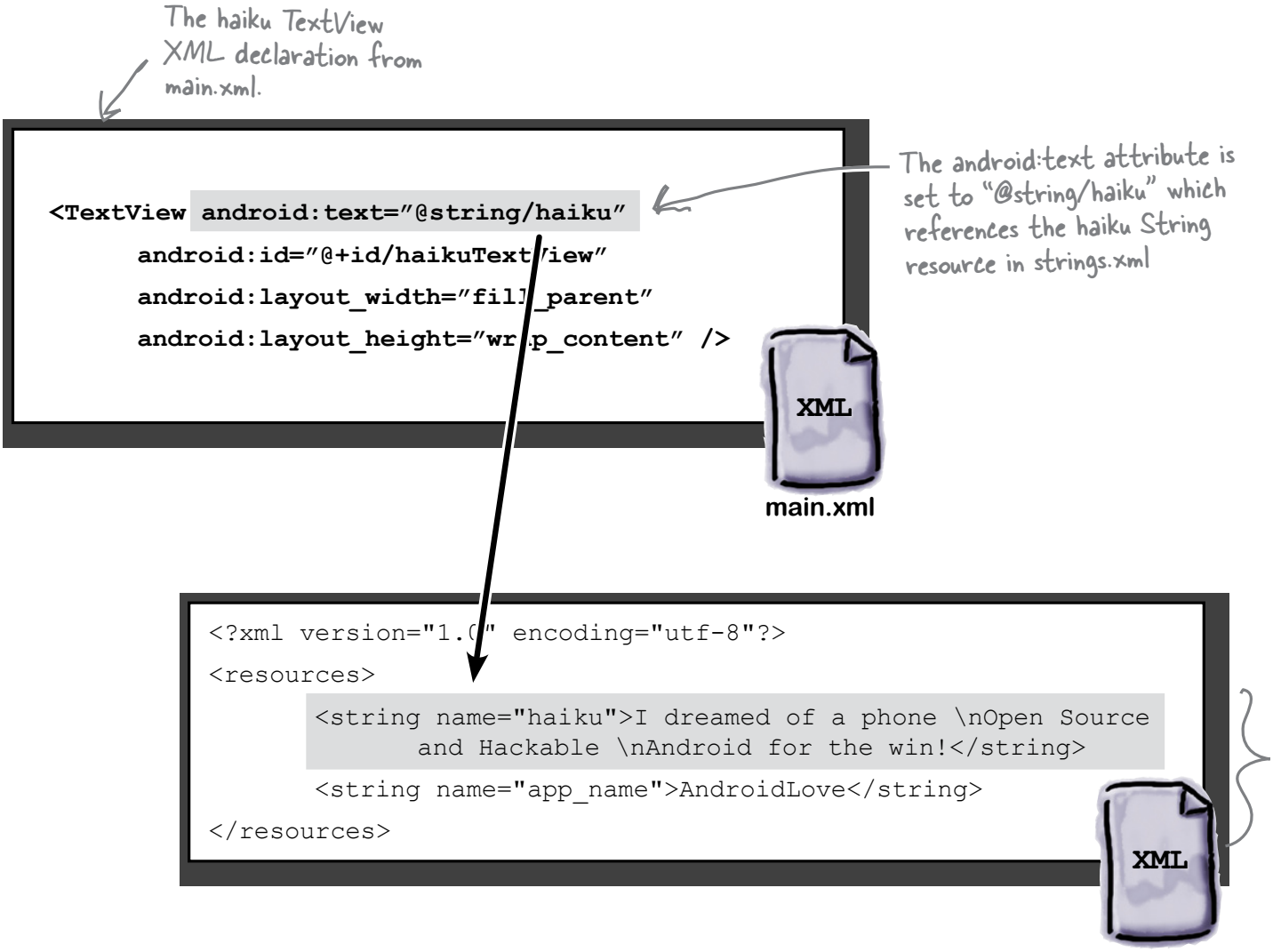
متن دکمه را تغییر بده

این خیلی خوبه که تو دکمه را اضافه کردی اما روی دکمه "@+id/Button01" نوشته شده و این زیاد جالب نیست .
بزار ببینیم چطور باید اون را عوض کرد .



چرا متن روی دکمه چنین چیزی هست ؟

برای پی بردن به این Viewهایی که داریم را مقایسه میکنیم . بیا تا Text View و Button را مقایسه کنیم . به مشخصه Text هر دوتا View دقت کن .مشخصه Text شعرها به رشته ی haiku در string,xml اشاره میکنه .



حالا به مشخصات دکمه نگاه کن

مشخصه دکمه یعنی android:text پیشوند "@string/"
را ندارد . اونجا مقدار "@+id/Button01"نوشته شده .

The new Button XML declaration.

t attribute is referring to Button01.

```
<Button android:text="@+id/Button01"
        android:id="@+id/Button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

XML

main.xml

Wait a second! There is no Button01 string property in strings.xml. What gives?

جواب در پیشوند اون قرار داره ....

مقدار android:text در مشخصات TextView
به منبع رشته ای در فایل strings.xml اشاره دارد .

The answer lies in the prefix...

اما هیچ منبع رشته ای برای دکمه ما وجود ندارد !

The android:text property refers to a String resource in strings.xml.

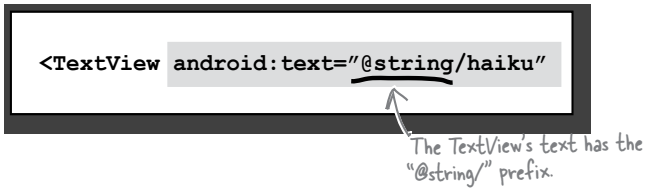
But there is no string resource for the Button!

پیشوند @string

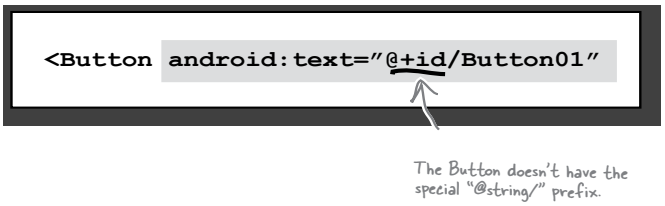
یک نگاه به مشخصات TextView شعرها بنداز و میبینی که اون یک پیشوند "@string/" دارد.

این پیشوند به خصوص به کامپایلر می گه که برای نوشته روی TextView، فایل string.xml را جستجو کنه و ما می بنیم که این پیشوند به خصوص در مشخصات دکمه دیده نمی شه و به همین خاطر هست که اون کار نمیکنه.

TextViewپیشوند "@string/" را دارد .



Button پیشوند "@string/" را ندارد .



there are no Dumb Questions

خپ اگه این طوره چرا خودم را با string.xml درگیر کنم ؟ بهتر نیست من هرچی که می خوام نشون بدم ستقیما توی همین لایه بنویسم ؟

از لحاظ تکنیکی بله ، اما این فکر خوبی نیست . منابع رشته ای برای این بوجود امدند که رشته های ثابت را از لایه های تو حذف کنند (مترجم : برای زیباتر شدن و خوانایی و... برنامه) . بهتر اونها را جدا از لایه ها نگه داریم و اندروید این کار را رای شما انجام میده.

پس به این خاطره که\روی دکمه متن @+id/button01 نوشته شده ؟

دقیقا

اگه دکمه پیشوند "@string/" را نداره پس چطور اصلا متنیرا نشون می ده؟

اگه اندروید پیشوند "@string/" را پیدا نکنه اون مستقیما همون چیزی را که در android:text وارد شده ، نشون میده.

منبع رشته ای را برای دکمه اضافه کن

برای این کار تو باید دو جای مختلف تغیراتی بدی . اول باید یک رشته جدید در فایل String.xml ایجاد کنی ، بعد از اون باید فایل main.xml را دستکاری کنی تا این رشته ای که بوجود اوردی را نمایش بده .

بزار با ایجاد یک رشته ی جدید شروع کنیم . string.xml را باز کن ، اینجا جاییه که باید رشته جدیدت را اضافه کنی و تو این کار را با ویرایش مستقیم فایل XML انجام میدی !

این کار را به شکل زیر انجام بده :

این قسمت را با نوشتن string شروع کن .string برای اندروید شناخته شده است و اون به معنی منبع رشته ای هست .

به اون یک اسم بده ، این اسم چیزی است که تو در کدهات استفاده میکنی تا متن خودت را نشون بدی

```
<string name="haiku">I dreamed of a phone \nOpen Source
        and Hackable \nAndroid for the win!</string>
```

این چیزی هست که در اخر نشون داده میشه

در این زیر محتوای فایل string.xml نشون داده شده . یک خصوصیت و مشخصه جدید اضافه کن و اسم اون را"love_button_text" بزار و مقدار اون را" Show me some Android love ! " قرار بده .

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
      <string name="haiku">I dreamed of a phone\nOpen Source
            and Hackable\nAndroid for the win!</string>
      <string name="app_name">AndroidLove</string>

      .....

</resources>
```

Add the new property here.

در این زیر محتوای فایل string.xml نشون داده شده . یک خصوصیت و مشخصه جدید اضافه کن و اسم اون را"love_button_text" بزار و مقدار اون را " Show me some Android love ! " قرار بده .

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="haiku">I dreamed of a phone\nOpen Source
        and Hackable\nAndroid for the win!</string>
    <string name="app_name">AndroidLove</string>
    <string name="love_button_text">Show me some Android love!</string>
</resources>
```

The element is a String element.

The element has a name attribute of "love_button_text".

And the value is set to "Show me some Android love!"

حالا کافیه فقط از اون استفاده کنی !

تا اینجا تو فقط منبع رشته ای را برای دکمه ات اضافه کردی . حالا وقتش رسیده تا این رشته را به دکمه خودت در فایل main.xml وصل کنی .

در زیر محتوای فایل main.xml قرار داره . حالا که تو منبع رشته ای به نام love_button_text داری

پس اون را در تعریف دکمه خودت قرار بده تا متن مورد نظرت را نمایش بده .

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

<Button android:id="@+id/Button01"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="”…….”
      />

<TextView android:text="@string/haiku"
      android:id="@+id/haikuTextView"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content" />

</LinearLayout>
```

از پیشوند /@string به اضافه نام منبع رشته ات اینجا استفاده کن تا دکمه ات متنی را که داخل منبع رشته ای هست را نشون بده .

در اینجا بیشوند /@string به اندروید میگه که love_button_text را داخل فایل string.xml

جستجو کنه و مقدارش را که همون show me android love! هست را نشون بده.

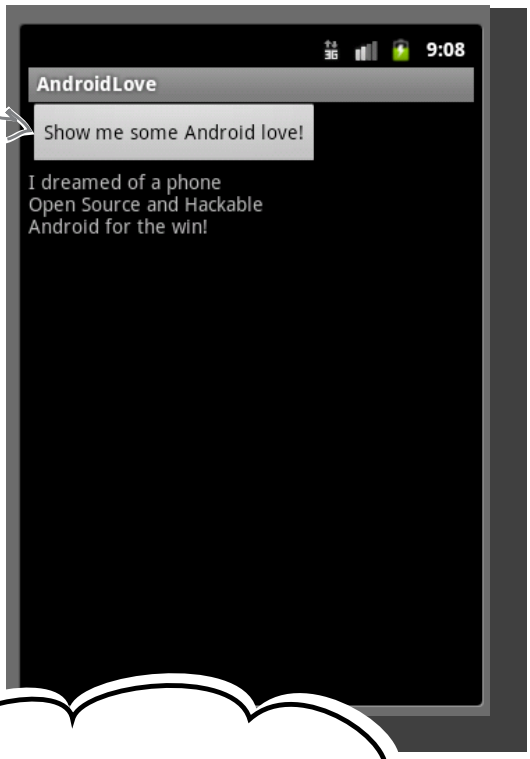
Test Drive

واوو ! تو دکمه ای اضافه کردی که متن عجیت غریبی داشت ، و تو برای این که اون را اصلاح کنی رفتی و یک منبع رشته ای ساختی ، و از اون منبع جدید در خصوصیت دکمه یعنی android:text استفاده کردی .

حالا ببینیم که این کارهایی که کردی جواب میدن یا نه !

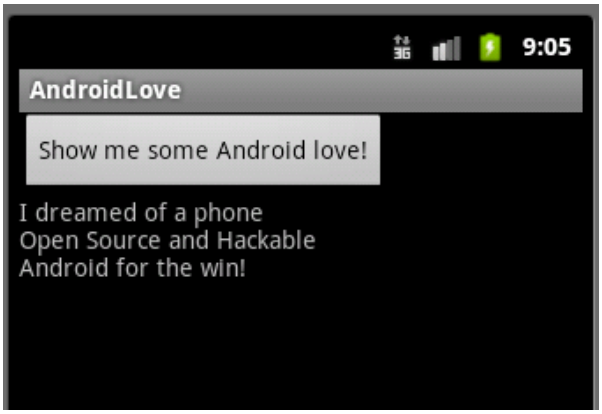
برنامه را دوباره اجرا کن ....

دکمه ما متنی را که داخل منبع رشته بود را به درستی نشون داد.



و میبینیم که کار کرد ! دکمه ما خیلی خوب شده !

خیلی خوبه ! این نشون میده که تو داری راه را درست میری . حالا نوبت این رسیده که متن شعرها را مخفی کنی .....



You need to hide this text.

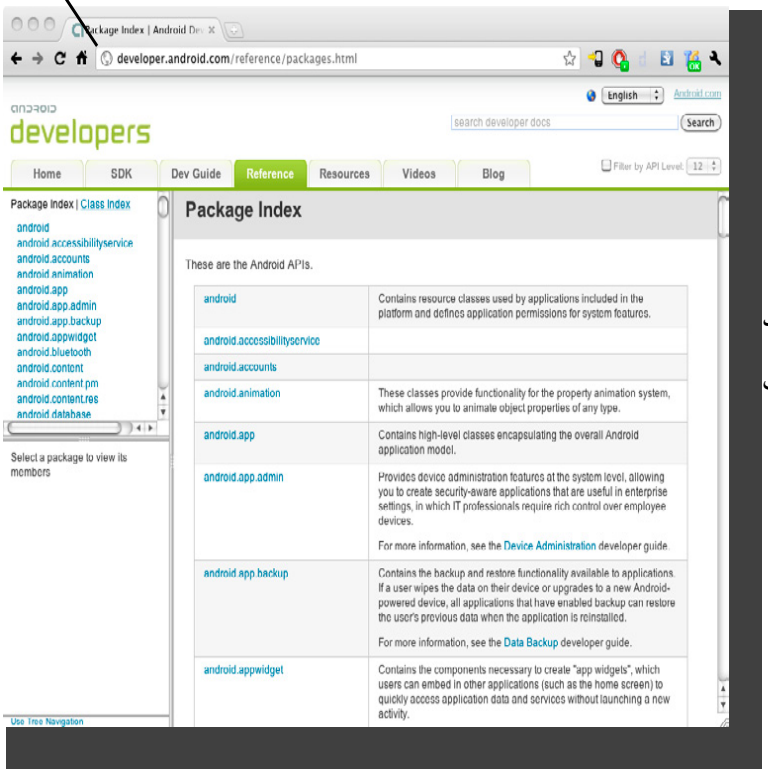
متن شعر ها را مخفی کن

حالا که دکمه ات را اضافه کردی و درست کار کرد ، حلا وقتش رسیده که بریم به مرحله بعد یعنی مخفی کردن متن شعرها.

تو این کار را چطوری انجام میدی؟

خپ ، ممکنه سریعا دوتا راه حل به ذهنت برسه ، اول این که متن را حذف بکنی و وقتی دکمه را فشار دادی اون بیاد و راه دوم این که اون متن را فقط غیب بکنی و بعد که دکمه فشار داده شده ظاهر بشه .

developer.android.com/reference



بزار گزینه غیب کردن متن را امتحان بکنیم !

بسیار خوب ، اما این که فقط بگیم متن را غیب میکنیم کمک زیادی نمی کنه درسته؟ تو باید بدونی دقیقا چطور این کار را انجام بدی . این چیز جدیدیه که تا حالا انجامش ندادی و باید بدونی که از کجا میشه در مورد چیزهای جدید در اندروید اطلاعات بدست بیاری . خوشبختانه ، اندروید یک وب سایت بزرگی مخصوص این کار داره !

تو میتونی اون را در ادرس زیر پیدا کنی

Developer.android.com/reference

(مترجم : میدونم ناراحت میشید اما باید واقعیت را بگم ..... متعصفانه و با کمال تاسف این سایت هم در لیست بلند و بالای تحریم های ایران قرار داره و شما نمیتونید به اون دسترسی پیدا کنید . اما از انجا که همیشه راه حلی هست ، شما میتونید از نرم افزارهایی که ادرس IP شما را عوض میکنن استفاده کنید تا به اون دسترسی پیدا بکنید یا ان که از افراد ماهر در برنامه نویسی اندروید کمک بگیرید من سایت www.kamalan.com را پیشنهاد میکنم.)

Documentation Navigation Up Close

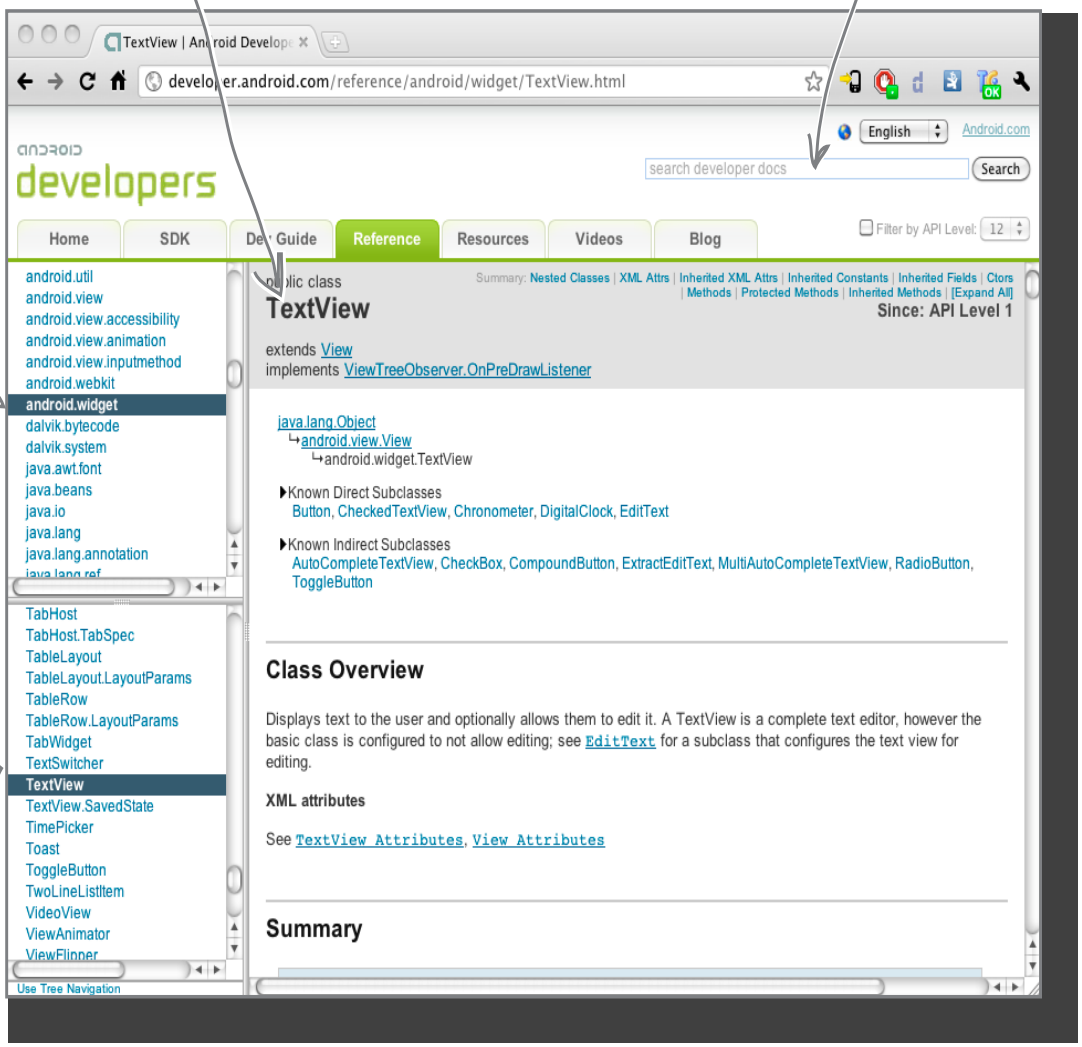
بزار یک چرخ سریعی در این وب سایت بزنیم تا شما با اون بیشتر اشنا بشین . شما میتونید با انجام یکی از این دو کار به چیزی که می خواهید برسید اول این که نام بسته و کلاسی را که میخواهید انتخاب کنید یا در کادر جستجو اسم کلاس خود تون را بنویسید . خپ از اونجایی که شما دنبال اضافه کردن چیز جدیدی به TextViewخودتون هستید شما باید به قسمت TextView نگاهی بیندازید.

اکه شما اسم کلاسی را که دنبالش هسنید را میدونید اما اسم بسته را نمی دونید شما میتونید اون را در کادر جستوجو بنویسد

وقتی که شما روی یکی از کلاسها کلیک میکنید پانل اصلی جزییات اون را به شما نشون میده

در این قسمت شما لیست کامل بسته ها را مشاهده

میکنید ، روی یکی از اونهای کلیک کنید تا داخل اون را ببینید

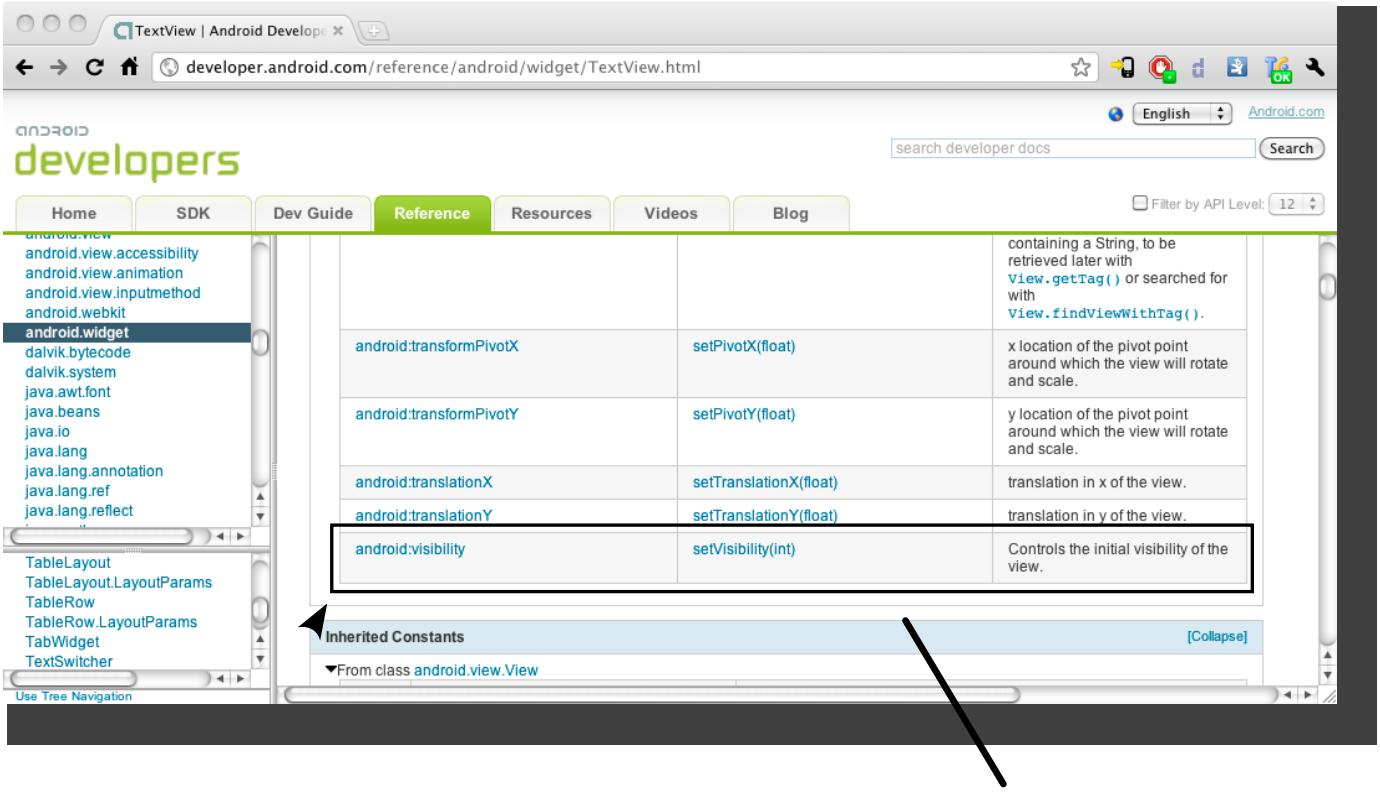


زمانی که یک بسته را انتخات کردید در این قسمت شما همه کلاسهای اون بسته را مشاهده میکنید

قسمت مشخصات XML را جستو جو کن

وقتی که شما در کلاس TextView چرخی بزنید متوجه میشید که این قسمت هم از تعدادی تابع جاوا و هم از تعدادی خصوصیت برای XML تشکیل شده . پس به این خاطره که TextViw خودش یک کلاس هست. اما چون الان شما میخواهید با فایل main.xml کار کنید که اون هم از کدهای XML ساخته شده پس روی قسمت XML تمرکز میکنیم .

ایا چیز قابل توجه ای اینجا میبینی ؟ تو دنبال چیزی هستی که متن را مخفی کنه ....



این به نظر خوب میاد!

این مشخصه میگه که اونی میتونه ظاهر view را کنترل کنه . این دقیقا همون چیزیه که تو میخوای !

با استفاده از این تو میتونی کاملا متنت را غیب کنی .

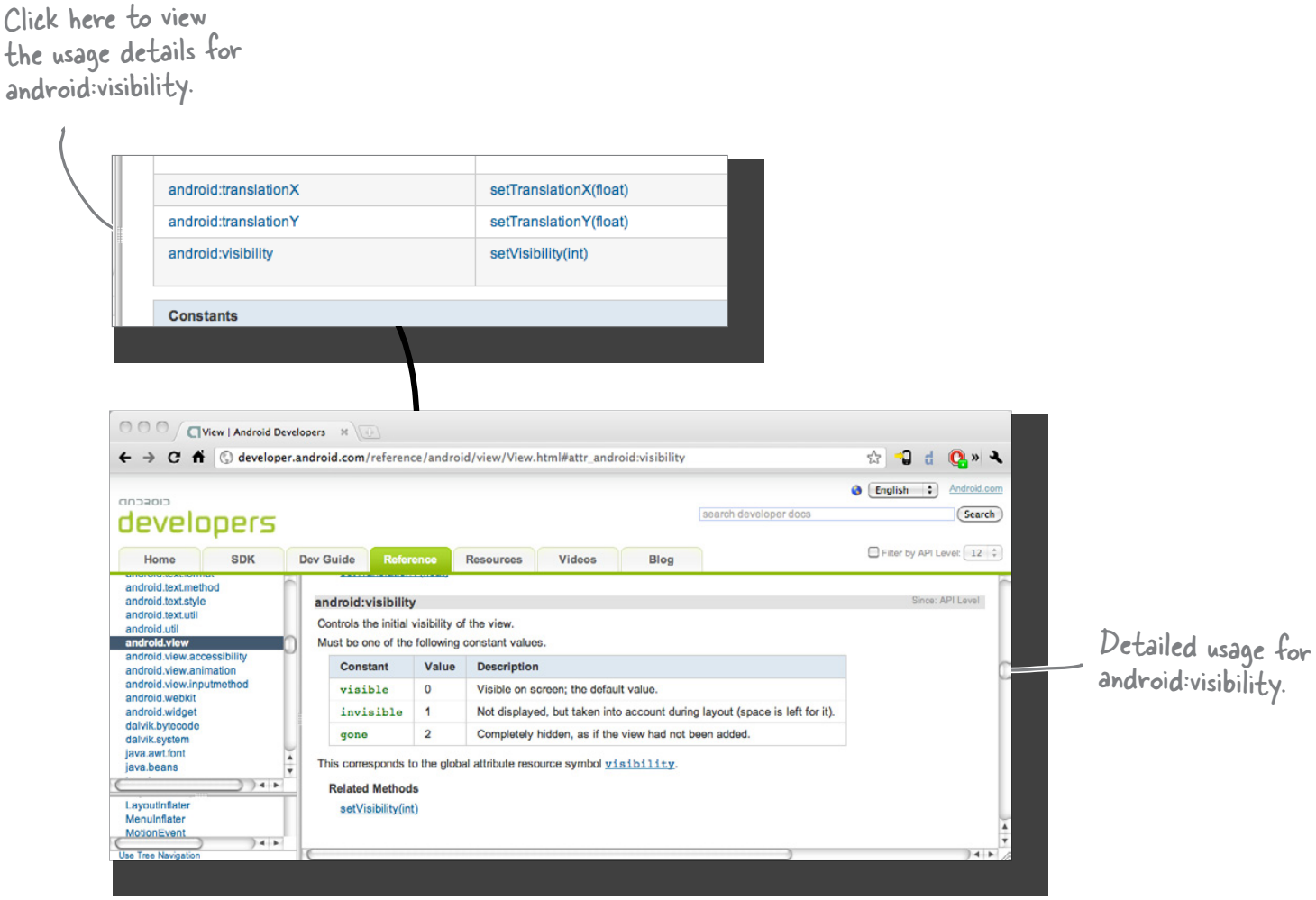
خپ حالا اون چطوری کار میکنه؟

جزییات این خصوصیت را ببین

اگه روی هر خصوصیتی که میخواهی کلیک کنی جزییات استفاده از اون نشون داده میشه .

حالا روی android:visibility کلیک کن و حالا جزییات اون را میبینی .

این قسمت به ما میگه که طرز استفاده از اون به این شکل هست :



مشخصات همیشه بین این علامتها هستند

```
android:visibility  =  "invisible"
```

این اسم خصوصیت ما هست

از عبارت invisible استفاده کنید چون میخواهید اون را غیب کنید

View **یک کلاس پایه هست که سایر ویجت ها از اون ارث بری میکنن**

View.java یک کلاس پایه هست که چندین تابع ، خصوصیت و ثابتهای مشترکی داره .

(مترجم : یعنی هر زیر کلاسی که از این کلاس پایه ارث بری کنه تمام این توابع و.... را هم ارث میبره ، مثل اموالی در دنیای ما به ورثه ها میرسه ، مفاهیم ارث بری و چند ریختی و..... از اساس و بنیانهای برنامه نویسی شی گرا هستند و جاوا هم یکی از زبانهای قدرتمند شی گرا هست . پیشنهاد می کنم برای کسب اطلاعات بیشتر و رهایی از سر در گمی های احتمالی کتابی که مفاهیم برنامه نویسی شی گرا را اموزش می ده را بخوانید)

و اگه شما به اول معرفی کلاسهای Button and TextView نگاه کنید میبینید که هردوی اونها از View ارث بردن .

در زیر کدهای لایه main.xml را میبینی . در این کدها android:visibility را بنویس

و مقدار اون را invisible قرار بده . این باعث می شه که شعرها مخفی بشن .

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

<Button android:text="@string/love_button_text"
        android:id="@+id/Button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

<TextView android:text="@string/haiku"
        android:id="@+id/haikuTextView"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"

        ..........................................................................
        />

</LinearLayout>
```

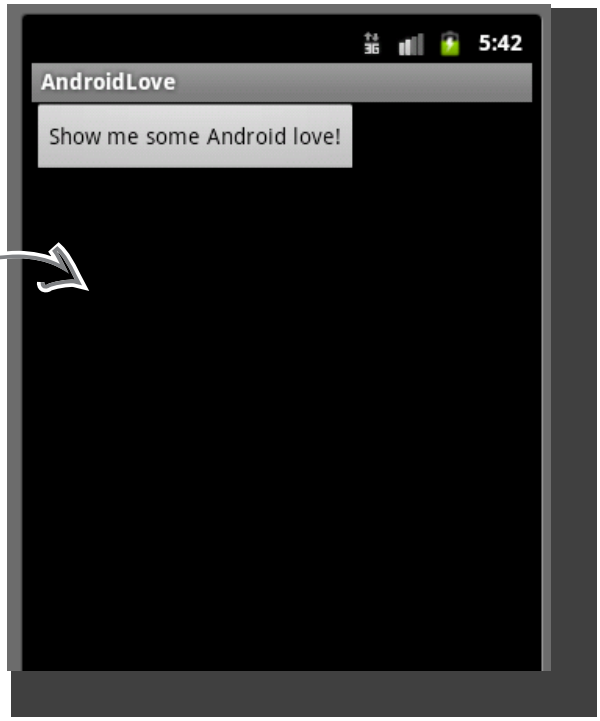
Android:visibility را اینجا اضافه کن

ما مقدار اون را نامریی یا invisible قرا ر دادیم که باعث می شه متن غیب بشه

Test Drive

تو با استفاده از خصوصیت android:visibility تونستی که TextView و شعرهای داخل اون را مخفی کنی . حالا بیا تا برنامه را اجرا کنیم تا مطمن بشیم که کار میکنه !

وقتی مقدار خصوصیت android:visibility را invisible قرار دادیم متن ما مخفی شد .



متن غیب شده . کارت عالی بود !

حیرت انگیزه ! تو اول دکمه را قرار دادی و بعدش هم متن را غیب کردی ، حالا فقط کاقی کاری کنیم که وقتی دکمه را فشار دادیم اون متن ظاهر بشه.

کاری کن که دکمه شعرها را نشون بده

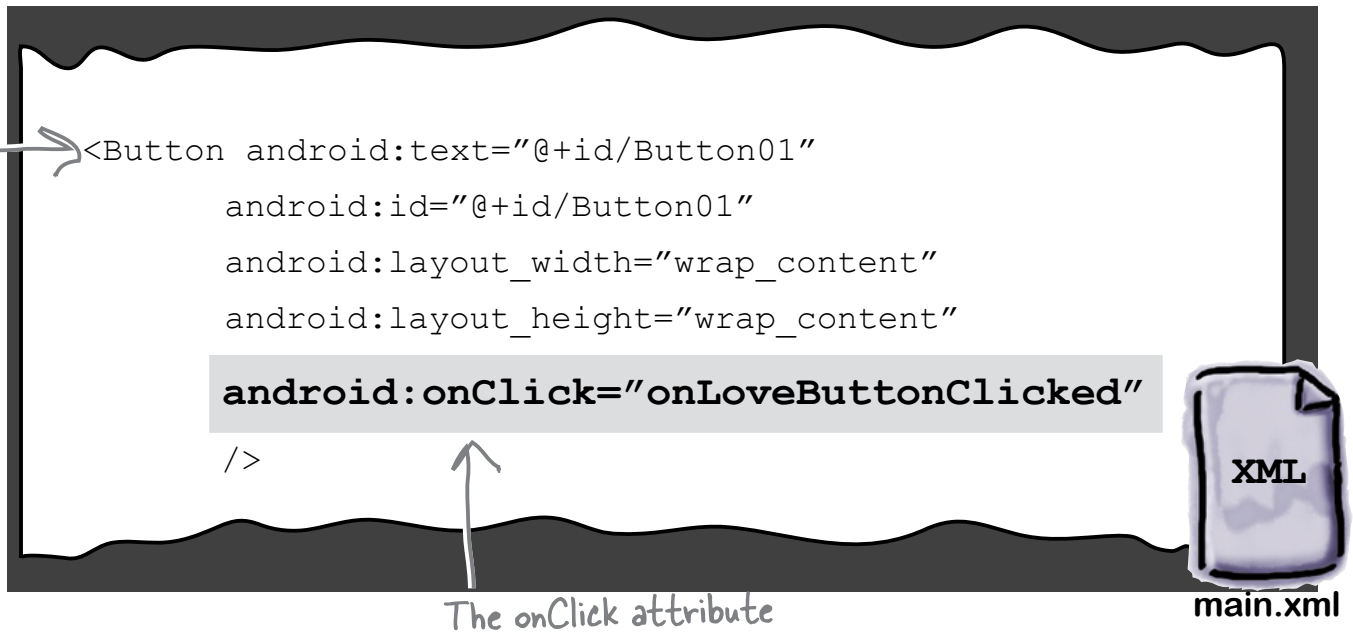
حالا وقتش رسیده که کاری کنیم دکمه کاری انجام بده ! button یک خصوصیتی داره که مخصوص این کاره و اسم اون هم android:onClick هست . مقداری که به این خصوصیت داده میشه همون اسم فعالیتی هست که می خواهی انجام بده .

خپ بیا تا از اون استفاده کنیم !

Android:onClick را به قسمت تعریف Button در فایل main.xml اضافه کن و به اون مقدار onLoveButtonClicked را نسبت بده . این اسم اختیاریه و شما هر چیزی میتونید بنویسید اما بهتره معرف چیزی باشه که می خواهد انجام بده .

قسمت تعریف دکمه در

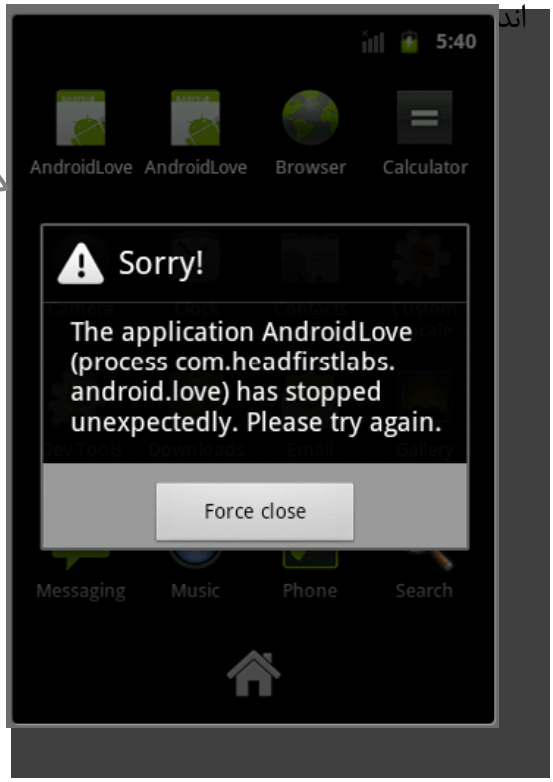
فایل main.xml



در حقیقت این یک تابع زبان جاوا هست که هنوز اون را ننوشتیم ...

تا الان تو صفحه برنامه را بروز رسانی کردی ، یک View جدید به اون اضافه کردی ، منابع رشته ای را اصلاح کردی و چیزهای جدیدی به اون اضافه کردی . همه این کارهایی که کردی فقط همون اول شروع برنامه خودشون را نشون میدن . اما برای دکمه تو باید کاری کنی که وقتی برنامه در حال اجراست کاربر بتونه اون کار و تغییر به خصوص را انجام بده و این همون اضافه کردن فعالیت و رفتار به برنامه هست . و اگه بخوای این کار را توی اند

اگه همین الان برنامه را اجرا کنی و دکمه را فشار بدی اون وقت یک خطا دریافت میکنی . چون تو هنوز تابع onLoveButtonClicked را تعریف نکردی .



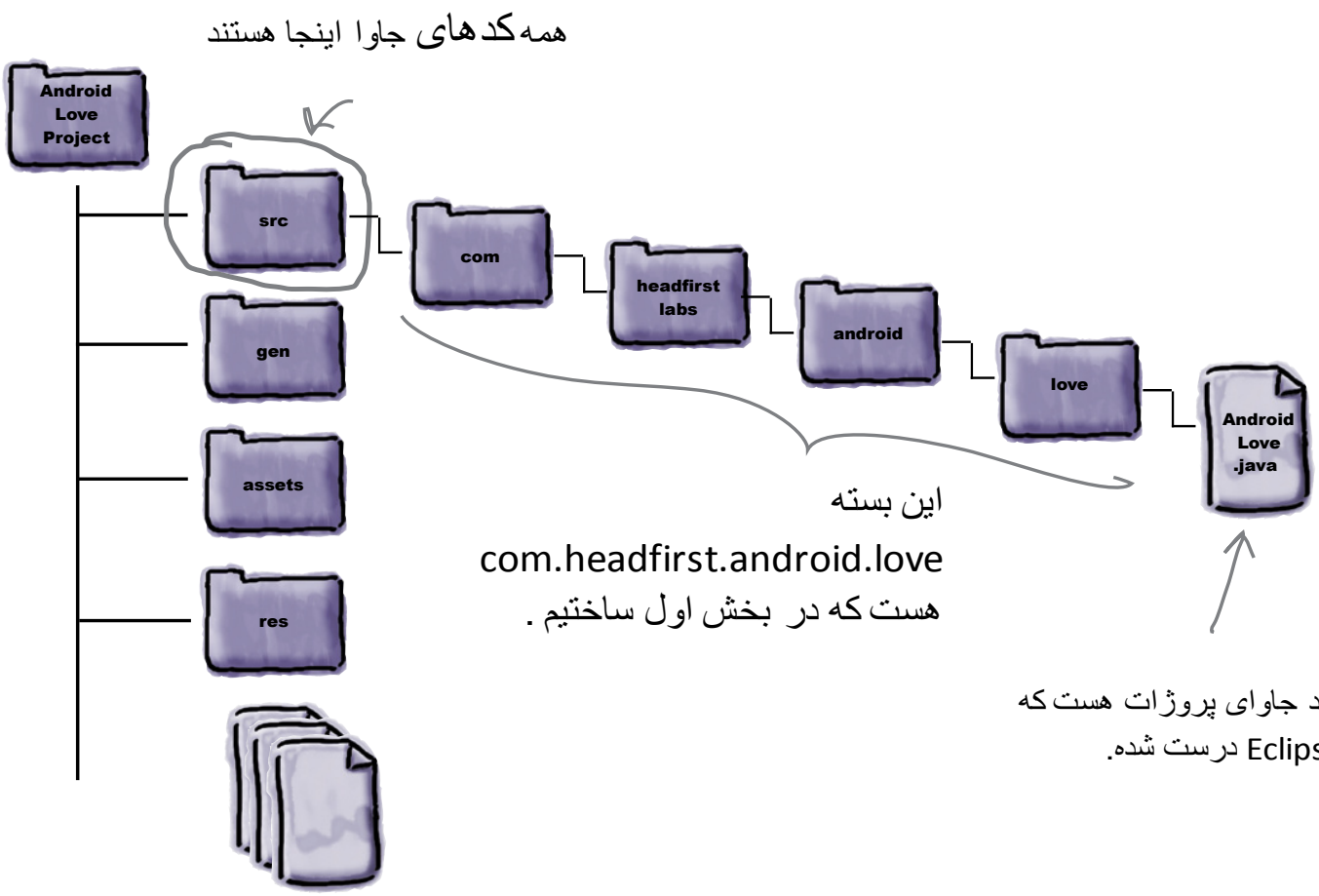
خب اگه این طوره بریم و تابع onLoveButtonClicked را تعریف کنیم ...

تعریف تابع onLoveButtonClicked

بنابراین تا اینجا فهمیدیم که نوشتن onLoveButtonClicked در خصوصیت android:onClick

یک تابع را در کدهای جاوا صدا میزنه . اما این تابع کجا باید باشه؟

بزار تا یک نگاهی به کدهای جاوا و محتویاتش در پروژه بندازیم .



ایا فقط یک فایل جاوا توسط Eclipse ساخته شده ؟

بزار تا از نزدیک یک نگاهی به اون بندازیم ...

اکتیویتی androidLove

(مترجم : وقتی به کلمه activity رسیدم ، گفتم خدایا این را دیگه چی معنی کنم ، بنویسم فعالیت ، کار ، بخش ..... گفتم ممکنه خواننده گیج بشه بنابراین اون را به زبان فینگلیش نوشتم که میشه اکتیویتی
اما اکتیویتی چیه ؟ اکتیویتی به طور مختصر همون چیزهایی هست که شما روی صفحه میبینید پس یک برنامه میتونه چندین اکتیویتی داشته باشه که در فصلهای اینده همین کتاب به اون اشاره شده .)
androidLove یک زیر کلاس از کلاس از پیش ساخته شده ای هست که اسم اون activity هست.
فکر کنید اکتیویتی یک کد جاوا برای نمایش صفحه است . و در حقیقت در این مورد به خصوص androidLove هم یک اکتیویتی هست که محتویات فایل main.xml را روی صفحه نشون میده .
بر روی فایل androidLove.java دو بار کلیک کنید و Eclipse به طور خورکار این فایل را در ویرایشگر باز میکنه .

کدهای androidLove.java

```
public class AndroidLove extends Activity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

androidLove از کلاس پایه activity ارث بری می کند

این کد به خصوص محتویات فایل main.xml را روی صفحه نمایش میده . به زودی میبینی که اون چطور کار میکنه !

دکمه ما انتظار داره که تعریف تابع onLovebuttonClicked **را در این کلاس یعنی** androidLove **پیدا کنه .**

چون androidLoveActivity مسولیت نمایش محتویات فایل main.xml را بر عهده داره ، اندروید سعی میکنه تعریف تابعی را که در خصوصیت android:onClick نوشتیم را در این کلاس یا اکتیویتی پیدا کنه .

اندروید دنبال کدی با قالب زیر در این کلاس می گرده :

این متد ارگومانی از نوع View باید دریافت کنه . که در اینجا همون دکمه ماست .
(چون دکمه هم یک زیر کلاس از کلاس پایه View هست)

```
public void onLoveButtonClicked ( View view )
```

نام تابع باید با نامی که در خصوصیت android:onClick نوشتیم مطابقت داشته باشه

تابع مربوط به فعالیت را اضافه کن

بیا تا تابع onLoveButtonClicked را به androidLove اضافه کنیم . وقتی این کار را انجام دادی و برنامه را اجرا کردی دیگه خطایی دریافت نمیکنی .

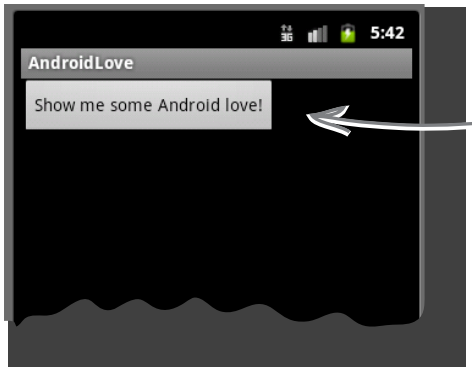
```
public class AndroidLove extends Activity {

      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
      }

      public void onLoveButtonClicked(View view) {
            //doesn't do anything yet
      }

}
```

The new onLoveVuttonClicked method that's referenced from the android:onClick Button attribute.

AndroidLove.java

Test Drive

حالا برنامه را اجرا کن و دکمه را فشار بده . در حال حاضر این دکمه کاری انجام نمیده اما می بینی که خطا هم نمی ده .



You can run the app and click the button now. Nothing will happen, but the app won't force close either.

پیاده سازی تابع مربوط به فعالیت

تا تا الان خوب کار کردی ! ما برای دکمه یک تابع در خصوصیت android:onClicked تعریف کردیم که برای ما کار انجام بده. (onLoveButtonClicked) تابع ما هیچ پیاده سازی ای نداره (مترجم: یعنی هیچ کدی ننوشتیم که بگیم چه کاری انجام بده) ، اما با این حال چون ما اون را تعریف کردیم ، وقتی دکمه را فشار میدیم برنامه متوقف نمی شه . وواوو !

حالا دیگه وقتش رسیده تا این تابع را پیاده سازی کنیم و کاری کنیم که متن شعرها را نشون بده !

در حقیقت پیاده سازی تابع onLoveButtonClicked شامل دو قسمت هست . اولش باید یک جوری اون را به TextView وصل کنی و بعد باید خصوصیت visibility اون را true قرار بدی . ساده به نظر میاد این طور نیست ؟

فوق العاده است ! بزار تا شروع کنیم ....

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
}

public void onLoveButtonClicked(View view) {
        TextView haikuTextView =    ?!?
}
```

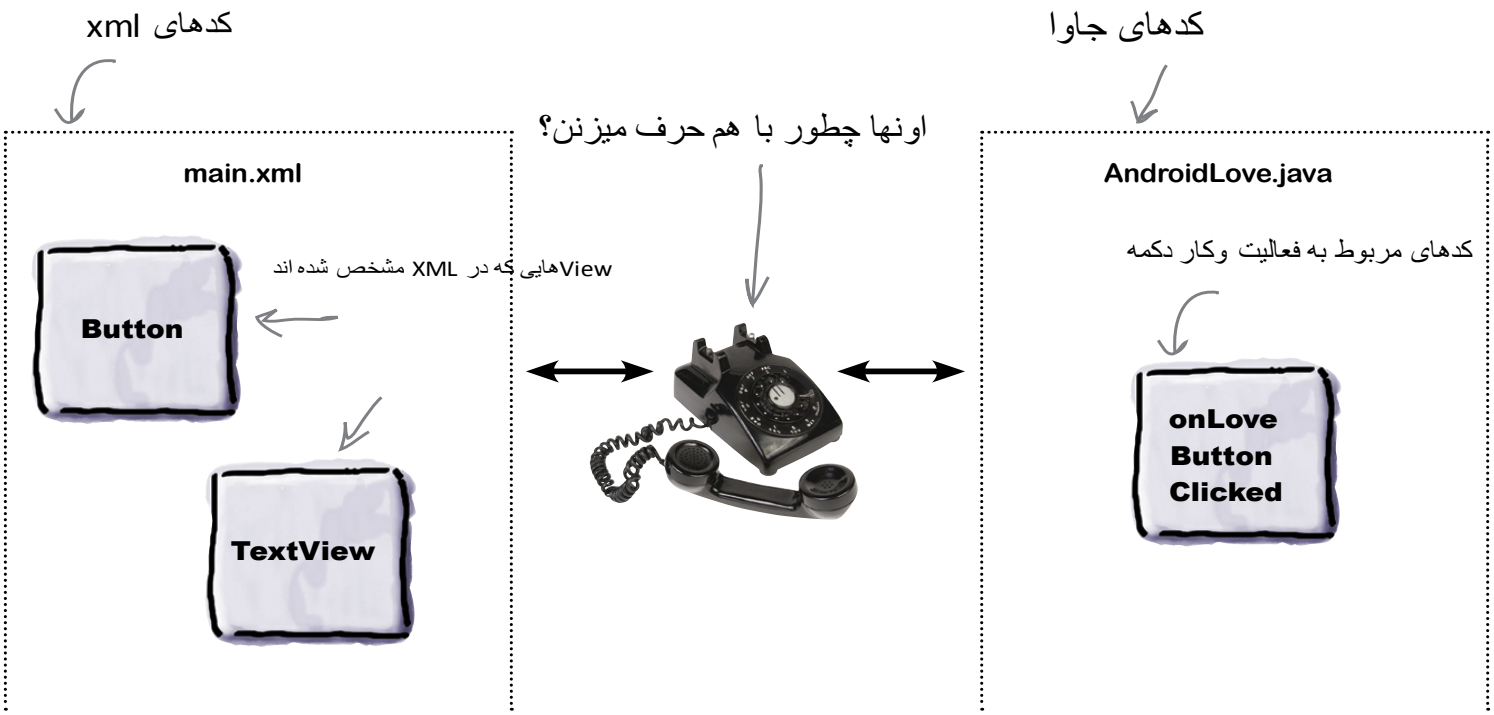
یک متغیر بساز که به Textview مربوط به شعر ها اشاره کنه ....

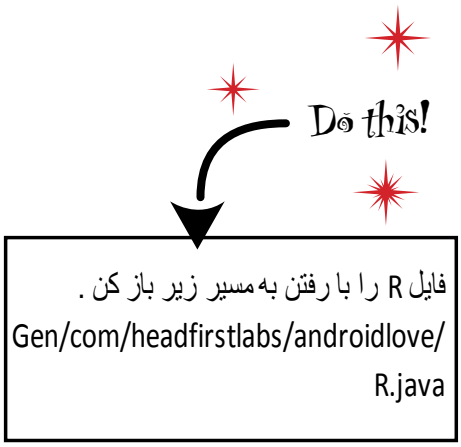
AndroidLove.java

صبر کن ، تو چطوری متغیرت را به TextView شعرها وصل می کنی ؟

از XML به JAVA

تو الان وصل نیستی . (دکمه و TextViewای که شعرها را نشون میده در فایل main.xml مشخص شده اند) . اما کدهای مربوط به کار و فعالیت دکمه در فایل جاوا قرار دارن . تو چطور می خوای Viewهایی که در فایل XML قرار دارن را به کدهای جاوا وصل کنی ؟





فایل "R"

برای حل مشکل ارتباط این دوتا اندروید فایلی ساخته که فایل "R"صداش میزنن . این فایلی هست که حاوی ثابتهایی است که میتونه جاوا را به Viewهایی که در main.xml هستند وصل کنه . راستش را بخوایی تو میتونی با استفاده از اون تو میتونی با استفاده از اون به تمامی منابعی که برای برنامه ات تعریف کردی دسترسی داشته باشی ! منبع رشته ای را که ساختی یادت هست ؟ تو میتونی به اون هم وصل بشی .

فایل R شامل تعدادی ثابت به شکل public static final هست که هر کدام از اونها به یک چیز در فایل XML اشاره میکنن. این ثابتها در قالب interfaceهایی بر اساس فایلهای XML دسته بندی شدن.
(مترجم: اگه میپرسید Public static final چیه، حتما یک کتاب در مورد زبان جاوا بخوانید چون هر چی جلوتر میریم از زبان جاوا بیشتر استفاده میکنیم چون برنامه نویسی اندروید با زبان جاوا هست (البته توی این کتاب) و باید حتما این زبان را بلد باشین.)

Interfaces grouping the constants.

Constants referring to XML resource.

```
public final class R {
        public static final class attr {}
        public static final class drawable {
                public static final int icon=0x7f020000;
        }
        public static final class id {
                public static final int Button01=0x7f050000;
        }
        public static final class layout {
                public static final int main=0x7f030000;
        }
        public static final class string {
                public static final int app_name=0x7f040001;
                public static final int haiku=0x7f040000;
                public static final int love_button_text=0x7f040002;
        }
}
```

R.java

اندروید خودش تعدادی تابع داره که به شما کمک میکنه از این ثابتها استفاده کنید. یک نگاه به تابع onCreate در فایل androidLove.java بندازین. تابع setContentView یک ثابت R.java به عنوان ارگومان دریافت کرده که این ثابت مربوط به main.xml هست.

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
}
```

تابع serContentView ثابتی به اسم R.layout.main دریافت کرده که لایه ای که در فایل main.xml مشخص شده را روی صفحه نشون میده.

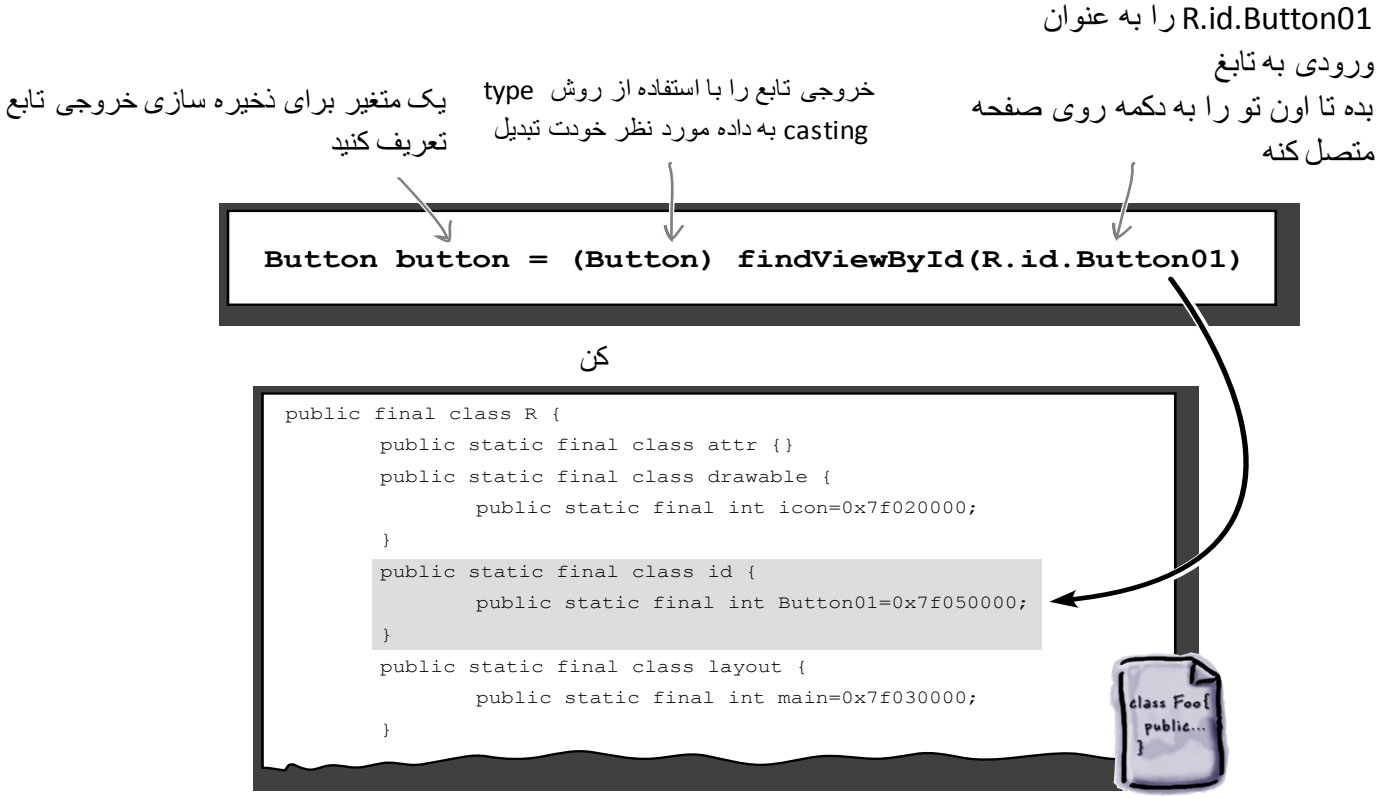
AndroidLove.java

وصل شدن به Viewها

تو تنها کاری که باید انجام بدی اینه که فقط با فایل R کار کنی ،
اما برای این کار هم باید به این فایل وصل بشی ! بسیار خوب ، اندروید
یک تابع دیگه ای هم داره که در این کار به تو کمک میکنه ، اسم این
تابع findViewById هست . تابع findViewById در کلاس پایه activity
قرار داره پس تنها کاری که باید بکنی اینه که از اون استفاده بکنی (یعنی نیازی به پیاده سازی از طرف کار بر
نداره) چون کلاس androidLove از کلاس activity ارث بری میکنه .

این تابع یک ارگومان از فایل R دریافت میکنه که همون ثابتی هست که مربوط به View مورد نظر است .
اما چون این تابع برای همه Viewها طراحی شده و کلی هست ، اون یک View کلی را برمی گردونه نه یک
ریز کلاس خاص از View (مثل دکمه ، TextView و .....) . ولی حل این مشکل کار بسیار ساده ای هست ،
فقط کافیه که خروجی تابع را با روش type casting به خروجی مورد نظر خودمون تبدیل کنیم .
بزار ببینیم اون چطور به دکمه روی صفحه وصل میشه .

(مترجم : در این کتاب از واژه reference که به معنی ارجاع است ، استفاده شده که من اون را وصل شدن ،
و در جایی که برای نمونه های ساخته شده از کلاسهای مختلف استفاده شده اون را متغیر ترجمه کردم)



به TextView یک شناسه نسبت بده

یک نگاه دیگه به فایلR.java بنداز . ثابتی برای دکمه وجود داره اما ثابتی برای TextView مربوط به شعرها نیست . **عجیبه، مگه نه** ؟

مسله اینجاست که ثابتهایی که در فایل R برای Viewها تولید میشن بر اساس خصوصیت android:id در فایلmain.xml هستند .

```
<Button android:text="@string/love_button_text"
        android:id="@+id/Button01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onLovebuttonClicked" />

<TextView android:text="@string/haiku"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:visibility="invisible" />
```

این خصوصیت اندروید نام ثابتی را که در فایلR تولید می شه را مشخص میکنه .

هیچ android:id ای برای TextView مشخص نشده بنابراین هیچ ثابتی هم در فایلR ایجاد نخواهد شد.

main.xml

هیچ android:id ای در فایل main.xml برای TextView مشخص نشده ، بنابراین ثابت فایل R هم تولید نمیشه . نگران نباش تو میتونی خودت اون را اضافه کنی ، در زیر تو کدهای فایل main.xml را میبینی خصوصیت android:id را به اون اضافه کن و اسم اون را "haikuTextView" بزار .

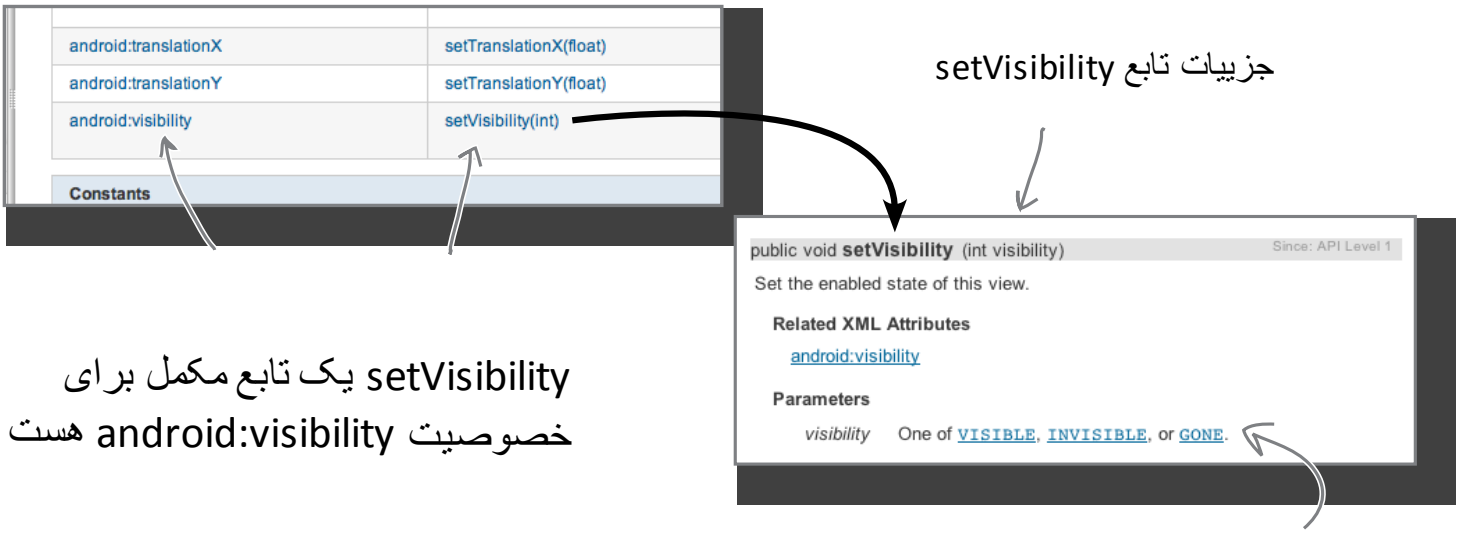
```
<TextView android:text="@string/haiku"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:visibility="invisible"

        ..........................................................
/>
```

هیچ android:id ای در فایل main.xml برای TextView مشخص نشده ، بنابراین ثابت فایل R هم تولید نمیشه . نگران نباش تو میتونی خودت اون را اضافه کنی ، در زیر تو کدهای فایل main.xml را میبینی خصوصیت android:id را به اون اضافه کن و اسم اون را "haikuTextView" بزار .

```
<TextView android:text="@string/haiku"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:visibility="invisible"
        android:id="@+id/haikuTextView"
/>
```

تابع جاوای مکمل

اکثر مشخصاتی که تو میتونی از طریق کدهای XML اونها را مشخص کنی از طریق کدهای جاوا هم قابل دستکاری هستند . این مهمه چون تو میخوای شعرها را از طریق کدهای جاوا نمایش بدی . بزار یک نگاه دیگه ای به مستندات android:visibility بندازیم و دنبال یک تابع مکمل برای اون باشیم .



جزییات تابع setVisibility

setVisibility یک تابع مکمل برای خصوصیت android:visibility هست

این ثابتها در کلاس پایه Activity هستند

قطعات را کنار هم بچین

تو همه تیکه هایی که لازمه تا تابع androidLoveButtonClicked را تکمیل کنی ، جمع کردی .
در زیر تو کدهای اکتیویتی androidLoveَ را میبینی که برای تابع androidLoveButtonClicked
چیزی نوشته نشده . قطعات زیر شامل همه کدهایی هستند که برای کامل کردن تابع به اونها نیاز داری .
از این قطعات استفاده کن و پیاده سازی تابع را کامل کن .

```
public class AndroidLove extends Activity {

	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
	}

	public void onLoveButtonClicked(View view) {
	.....

	.....

	.....

	}
}
```

AndroidLove.java

قطعات را کنار هم بچین

تو همه تیکه هایی که لازمه تا تابع androidLoveButtonClicked را تکمیل کنی ، جمع کردی .
در زیر تو کدهای اکتیویتی androidLove را میبینی که برای تابع androidLoveButtonClicked چیزی نوشته نشده . قطعات زیر شامل همه کدهایی هستند که برای کامل کردن تابع به اونها نیاز داری .
از این قطعات استفاده کن و پیاده سازی تابع را کامل کن .

```
public class AndroidLove extends Activity {

      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.main);
      }

      public void onLoveButtonClicked(View view) {
          TextView textView = (TextView) findViewById(
                R.id.haikuTextView );
          );

          textView.setVisibility( View.VISIBLE );

      }
}
```

Get the TextView reference using the R constant.

Set the TextView visibility to true so it's displayed.

AndroidLove.java

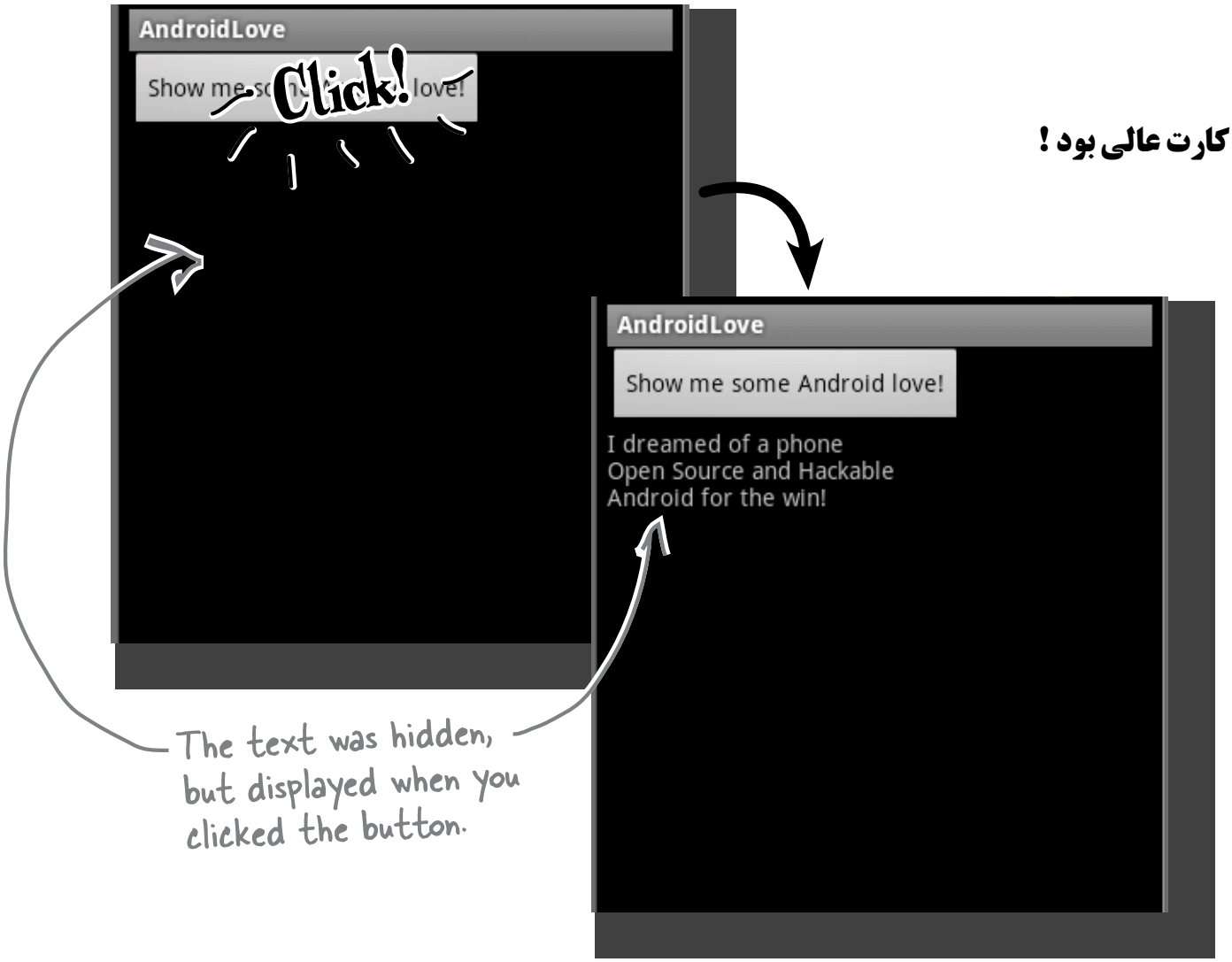
Test Drive

حالا که پیاده سازی تابع تموم شد برنامه را اجرا کن تا نتیجه را ببینی .

تو انجامش دادی !

وقتی تو این فصل را شروع کردی برنامه
androidLove کار ای انجام نمی داد .
اما حالا تو کاری کردی که
چیزی را انجام بده ! و برای محقق شدن اون
تو یک View جدید اضافه کردی و از یک منبع
رشته ای جدید برای اون استفاده کردی ، کدهای
اون را در جاوا ساختی و از فایل R استفاده کردی
تا بین جاوا و xml ارتباط برقرار کنی .

کارت عالی بود !



جعبه ابزار مربوط به اضافه کردن رفتار

حالا که به طور کامل به دکمه ات واکنش و فعالیت اضافه کردی، تو دیگه میتونی با همه برنامه هات این کار را بکنی !

از ویرایشگر گرافیکی استفاده کن تا به اسونی View جدیدی اضافه کنی .
اگه نیاز داشتی منبع رشته ای جدیدی اضافه کن از پیشوند@string استفاده کن تا به منبع رشته ای وصل بشی .
از مستندات اینترنتی استفاده کن تا از همه خصوصیاتی که میتونی در فایل XML استفاده کنی اگاه بشی .
اگه میدونی که دنبال چی هستی اما نمیدونی کجا پیداش کنی از کادر جستوجو استفاده کن .
به Viewها با استفاده از تابع findViewById وصل شو و ثابتهایی که در فایل R هست را به عنوان ارگومان به اون بده .
اکه میخوای از تابع findViewById استفاده کنی مطمن شو که از خصوصیت android:id در View مورد نظرت استفاده کردی .
برای اضافه کردن فعالیت به دکمه ات از android:onClick استفاده کن .این تابع را در همون اکتیویتی پیاده سازی کن که لایه کلی را روی صفحه قرار داده .
یادت باشه که تمام منابع جاوا در پوشه /src قرار داره .

درست کردن فعالیت برای دکمه

از خصوصیت onClick استفاده کن تا نام تابع مربوط به فعالیت را مشخص کنی .
اکتیویتی را که دکمه در اون قرار داره باز کن .
در این اکتیویتی تابعی اضافه کن که با نامی که در خصوصیت android:onClidk نوشتی برابر باشه .
مظمعن شو که این تابع یک ارگومان از نوع view به عنوان پارامتر یا ورودی دریافت میکنه .