Supporting Different Densities

This lesson shows you how to support different screen densities by providing different resources and using resolution-independent units of measurements.

THIS LESSON TEACHES YOU TO

YOU SHOULD ALSO READ

- Supporting Multiple Screens
- Icon Design Guidelines

TRY IT OUT

Download the sample app

NewsReader.zip

Use Density-independent Pixels

One common pitfall you must avoid when designing your layouts is using absolute pixels to define distances or sizes. Defining layout dimensions with pixels is a problem because different screens have different pixel densities, so the same number of pixels may correspond to different physical sizes on different devices. Therefore, when specifying dimensions, always use either dp or sp units. A dp is a density-independent pixel that corresponds to the physical size of a pixel at 160 dpi. An sp is the same base unit, but is scaled by the user's preferred text size (it's a scale-independent pixel), so you should use this measurement unit when defining text size (but never for layout sizes).

For example, when you specify spacing between two views, use dp rather than px:

```
<Button android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/clickme"
    android:layout_marginTop="20dp" />
```

When specifying text size, always use sp:

```
<TextView android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="20sp" />
```

Provide Alternative Bitmaps

Since Android runs in devices with a wide variety of screen densities, you should always provide your bitmap resources tailored to each of the generalized density buckets: low, medium, high and extra-high density. This will help you achieve good graphical quality and performance on all screen densities.

To generate these images, you should start with your raw resource in vector format and generate the images for each density using the following size scale:

- xhdpi: 2.0
- hdpi: 1.5
- mdpi: 1.0 (baseline)
- ldpi: 0.75

This means that if you generate a 200x200 image for `xhdpi` devices, you should generate the same resource in 150x150 for `hdpi`, 100x100 for `mdpi` and finally a 75x75 image for `ldpi` devices.

Then, place the generated image files in the appropriate subdirectory under `res/` and the system will pick the correct one automatically based on the screen density of the device your application is running on:

```
MyProject/
  res/
    drawable-xhdpi/
        awesomeimage.png
    drawable-hdpi/
        awesomeimage.png
    drawable-mdpi/
        awesomeimage.png
    drawable-ldpi/
        awesomeimage.png
```

Then, any time you reference `@drawable/awesomeimage`, the system selects the appropriate bitmap based on the screen's dpi.

For more tips and guidelines for creating icon assets for your application, see the Icon Design Guidelines (/guide/practices/ui_guidelines/icon_design.html).