# Unit 2 – Lesson 1. Introduction to DPI and ImageView

**Aim:**

How do we add images to our Android apps?

**Objectives:** After the lesson, students will be able to:

1. Obtain understanding of ImageView and @drawable folder
2. Understand qualities of images

**CLASS PROCEDURE:**

***Do Now:***

Find an image of some Android symbol on the internet. Save the image to your USB flash drive. Save the image in png or jpeg file format. Then add the image to an Android app project. How do we display the image on the app using ImageView?

1. To add an image, you first need to save the image in the formats of png, gif or jpg.
2. You will need to add the image to the Android Studio app > res > drawable folder. Here are the steps:
3. Right click on res, new **Image Asset**
4. On **Asset type** choose **Action Bar and Tab Icons**
5. Choose the **image path**
6. Give your image a name in Resource name

Next->Finish

1. Now you need to go to the activity\_main.xml, and add an ImageView to the layout file. Here’s the sample code:

<**ImageView  
 android:id="@+id/bugImageView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/droidImageView"  
 android:layout\_centerHorizontal="true"  
 android:contentDescription="@string/deitel\_logo"  
 android:src="@drawable/bug"** />

1. On Design view, check the position of the ImageView and test the “look” of the UI. Don’t forget to verify that the ID is unique.
2. Test your app and you shall see the image being displayed.

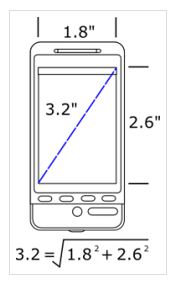
***Class Discussion:***

1. What is Pixel Density?

* The screen density is quoted as **Pixels Per Inch**, PPI, and is the number of pixels that fit into an inch.
* The higher the number then the sharper images look on the display, therefore consumers consider a high PPI figure an advantage when buying a device. Sometimes the figure is quoted as **Dots Per Inch**, DPI.

1. How is DPI being calculated?

* The DPI (or PPI) figure for a screen can be calculated given the resolution (number of x and y pixels) and size (diagonal measurement of the visible area) of a screen. Use the Pythagorean theorem to calculate the number of pixels in the diagonal then divide the result buy the screen size. i.e. PPI=(square root of x^2 + y^2)/screen size.



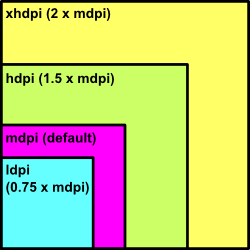
* For example for a HTC Wildfire S with 3.2″ at 320×480. Square the x (320\*320) to give 102,400. Square the height (480\*480) to give 230,400. Add the two squares together giving 332,800. Talk the square root to give 576, to the nearest integer. Then divide by the screen size (576/3.2) to give the screen density as 180 PPI. For the HTC ChaCha which has the same resolution screen but a a smaller screen size, 2.6″, the figure is 221 PPI (i.e. 576/2.6). The HTC Wildfire S was an upgrade to the HTC Wildfire. The original HTC Wildfire had 3.2″ screen at a resolution of 240×320, therefore for the density is square root(240^2+320^2)/3.2 = 125 PPI.

1. What is one of the biggest challenges for Android App developers?

* Android is a versatile OS with more than 1000 device manufacturers and more than 18000 distinct devices. Screen size of android phones vary from 2.6” – 6” and the resolution of screen ranges from 240 X 320 to 1440 X 2560 px with screen density from 120 to 640 dpi (ldpi to xxxhdpi).
* It is difficult for designers to create such designs which work well on all these devices irrespective of the size, density and aspect ratio of device and still stay developer friendly.

1. How can Android handle wide range of screen densities? What is mdpi, hdpi, xhdpi?

* Android can handle the wide range of screen densities by classing a display as either low density, medium density, high density or extra high density (the [DisplayMetrics](http://developer.android.com/reference/android/util/DisplayMetrics.html) class can be used in code to read the device density and screen size in pxels).
* The medium density setting is the reference point, at this point Android takes 160 PPI as the base line. If a device has a screen density around this figure (it does not need to be exactly 160 PPI, a 180 PPI screen could be medium density) a resource, such as a image file, is displayed at the original size, with no scaling. For lower and higher densities the image file may be scaled up or down so that it is displayed correctly on the screen.

[](http://tekeye.biz/wp-content/uploads/2012/02/androidscreenratios.png)

* When Android was first released the resources used in an Application (App) would be stored in a folder called **drawable.** To supported different screen densities four more folders were introduced: drawable-ldpi, drawable-mdpi, drawable-hdpi and drawable-xhdpi. These folders allow pre-scaled resources to be included in a App. This supports better quality interfaces because scaling a resource can introduce [pixelation](http://en.wikipedia.org/wiki/Pixelation) (images becoming ragged) or loss of detail. The scaling factors are 0.75 for low density, 1.5 for high density and 2 for extra high density. These factors mean that a low density screen is 120 PPI (0.75 \* 160 PPI), high density screen is 240 PPI (1.5 \* 160 PPI) and extra high density is 320 PPI (2 \* 160 PPI). The original drawable folder is still supported for backwards compatibility and it should just mimic the contents of the drawable-mdpi.
* The abbreviations are:
* LDPI – Low Dots Per Inch
* MDPI – Medium Dots Per Inch
* HDPI – High Dots Per Inch
* XHDPI – Extra High Dots Per Inch

***Pair – sharing Activity:***

Let’s start to prepare for our next project, Whack – a – mole. Find an image of a mole, and an image of a hole with the mole hiding, save both images to png format. Create a new Android project and name it “Whack-a-mole”. Place two buttons on the UI, and replace the background color of the buttons with the images. Complete the activity program so that when the app starts, it shows only the button with the mole. Upon the user clicks on the mole button, the button with the mole hiding should appear at the same place.

Hint: This tutorial will not give you the solution, but it might help you to get start! <http://www.mkyong.com/android/android-imagebutton-example/>