2. Change the background

We'll start making the watch face our own by giving it a custom background.

In this step, we will start making the watch face our own by giving it a background. If at any point you are confused by the concepts discussed here, please refer to the *2-background* module and see how these steps may be implemented.

Prepare your image

The first step is to prepare the image. You can select any photograph of your choice, but note that some images with tiny details may not scale well on a small watch display. Crop it to a square shape and resize it to *600 x 600* pixels. It can be in `jpg` or `png` format. The next step is to rename it to `custom_background` (Android needs underscore in place of space).

After you completed this, "right click" the `res/drawable` directory in Android Studio and select "Reveal in Finder" (for Mac), "Show in Files" (Linux) or "Show in Explorer" (for Windows). Copy your image file into the directory `res/drawable-nodpi`.

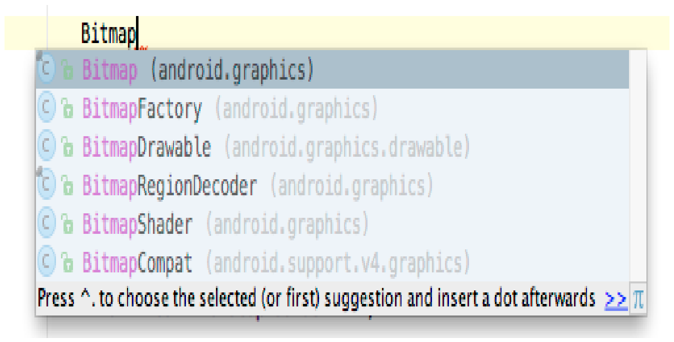
If you do not have an image, go to `res/drawable-nodpi` directory in *2-background* and copy the file `custom_background.jpg` to `res/drawable-nodpi` in *1-base*.

Create Bitmap and Paint objects

Before we can display the bitmap, we need to load and instantiate the object. Since we only want to do this once and we don’t need the dimension of the screen, we can put this into the onCreate method.

Pro-tip

We will need to import additional classes from Android framework / Java packages to help us throughout this code lab. As you type out class names like Bitmap, Android Studio will display a drop down list. Using the up and down arrow keys, you can select the class that you are meaning to type (see below). When you get to the correct one and press enter, Android Studio will autocomplete the name for you as well as adding an import line at the top of the file which brings the functionality of that class into use.


Bitmap
Bitmap (android.graphics)
BitmapFactory (android.graphics)
BitmapDrawable (android.graphics.drawable)
BitmapRegionDecoder (android.graphics)
BitmapShader (android.graphics)
BitmapCompat (android.support.v4.graphics)
Press ^. to choose the selected (or first) suggestion and insert a dot afterwards >>
If you missed this, don’t worry. The class name will be underlined in red if the import statement is not added. At this point, you can take the cursor back to that class name and press Alt-Enter and the import will be magically inserted.

Create a private `Bitmap` variable `mBackgroundBitmap` in the `MyWatchFaceService.Engine` class. This enables other methods to get hold of the `Bitmap` to resize it and draw later.

Within `onCreate`'s body in the `MyWatchFaceService.Engine` class, we will instantiate the bitmap like this:

```
mBackgroundBitmap = BitmapFactory.decodeResource(getResources(),
R.drawable.custom_background);
```

Resize the Bitmap object

Next we are going to resize the background `Bitmap`. Since dimensions are not available in `onCreate`, we resize the `Bitmap` in `onSurfaceChanged`.

In the `onSurfaceChanged` method, we add the code to resize the `Bitmap` object using the `width` of the screen (the third parameter of the `onSurfaceChanged` method) and the width of the background `Bitmap`:

```
float mScale = ((float) width) / (float) mBackgroundBitmap.getWidth();
```

Then we scale the background bitmap to match the screen's dimensions:

```
mBackgroundBitmap = Bitmap.createScaledBitmap(mBackgroundBitmap,
(int)(mBackgroundBitmap.getWidth() * mScale),
(int)(mBackgroundBitmap.getHeight() * mScale), true);
```

Draw the background

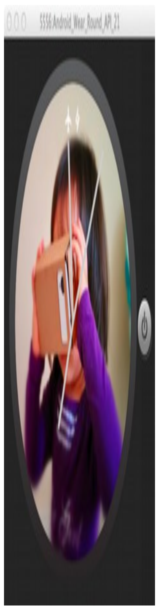
Now that the `Bitmap` is correctly sized - let’s draw it!

- Delete the `drawRect` code in `onDraw` which draws a black square and wipes the frame clean.
- Add code to draw the background bitmap in `onDraw`. We suggest you place this code directly below the code you deleted in the last step.

```
canvas.drawBitmap(mBackgroundBitmap, 0, 0, mBackgroundPaint);
```

Run the watch face again

In the first step, you learned how to install the watch face to your device or emulator. Now it's time to do that again! Your watch face should look something like this:



Summary

In this step you’ve learned about:

- Loading a bitmap object
- Resizing it based on the screen dimensions
- Drawing it on the screen!

Next up

Let’s refine the watch arms!