Barbello:
Fitness App for Android

User Guide

Version 2.1

Table of Contents

Revision History....3
Introduction....4
About The File....4
Installing Required Softwares....5
Opening Android Project....6
Renaming Application Package Name....7
Inserting Data to Database....10
Setting Up Admob....15
Customizing Application Color....19
Customizing Image Resources....20
Customizing App Content....21
Running the Application....22
Publishing Android App....23
Updating App Version....24
About the Author....25

Revision History

Author	Remarks	Version	Date
Taufan E.	Updated hole documentation content	2.0	14 September 2015
Cahaya P. Alam	Updated change log	2.1	11 November 2015

Introduction

Thank you for purchasing our item, Barbello (formerly known as Daily Workout App). If you have any questions that are beyond the scope of this user guide, please feel free to post the questions via pongodev support forum. You can register an account using your Envato username and Purchase Code Item of this item. Thanks so much!

Do not forget to rate this item if you think it is great. And also like pongodev Facebook page here and follow pongodev Twitter here to get the latest information about update and new items.

About The File

Barbello is a fitness app template for Android that suitable for fitness app or workout app. Barbello is designed with the latest design trend, Google Material Design that is simple and beautiful. It is easy to customize. Data of the app are stored in SQLite database within the application. You can insert sequence images so that the workout images will be generated into single animated image to make it easy for your app users follow the workout steps. Users can create their own workout program for each days.

If you are a workout trainer or just want to create fitness app? This app is perfect for you to start building your apps. It also integrated with two formats of Admob, banner and interstitial ad for monetizing purpose.

The complete features are below:

- Android Studio project
- Google material design
- SQLite database
- Run on smartphone and tablet
- Admob integration
- Animated image with sequence images

- Workout day program
- Count down timer
- Share to other apps
- Easy to customize

When you purchase this item you will get the following assets:

- App source code
- User guide

Change Log version 2.1:

- Build in latest Android Studio (1.4.1)
- Support Marsmallow (API 23)
- Update gradle with latest (2.4) and Plugins Version (1.3.0)
- Update additional libraries to the latest version
- Fix error issue when imported to Android Studio
- Fix other bugs..

Installing Required Softwares

As this app built using Android Studio, you need to download Android Studio first in order to be able to configure the app. Besides Android Studio, you also need to install the **latest version** of Java Development Kit (JDK).

You can download Java Development Kit (JDK) here and Android Studio here. Install JDK first on your computer and after finishing installing JDK, add java path to system variables if necessary. You can read an article about adding java path into system variables in our blog here. Now, you can install Android Studio, just follow the instruction until it finish installing.

Opening Android Project

Now, you have already installed all softwares that required to configure Android project. To open Android project in Android Studio, please make sure you're **have connection and using latest version of Android Studio and SDK.** Then follow the following steps:

1. Run Android Studio, you will see **Welcome to Android Studio** window.

2. Select **Open an existing Android Studio project** (Illustration 1) and go to the location of where you store your Android project. After that click **Choose** button.

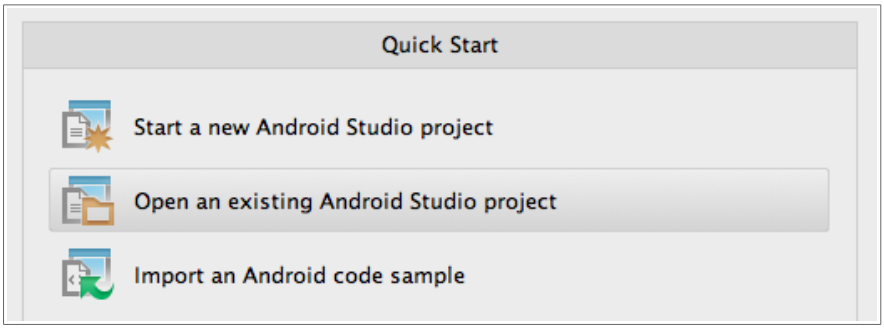


Illustration 1: Open Android project

3. A new window will open Android project and the project will appear on **Project** pane at the left side of the window (Illustration 2). Insure that your project is display in **Android** structure.

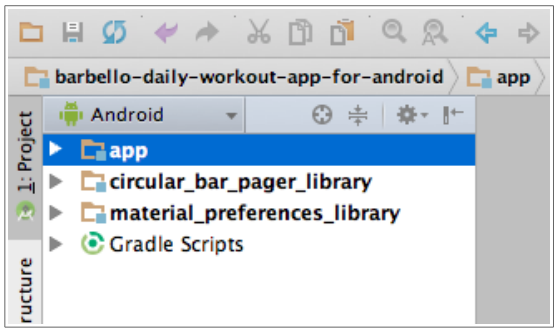


Illustration 2: Android project on Project window

4. You can now run your Android project by selecting **Run > Run 'app'** on menubar.

5. **Choose Device** window will appear (Illustration 3), you can select whether you want to run your app on Android device that connected to your computer or Android emulator that you have already created before. Click **OK** button to run the project. You can see this article for more information about how to run Android project on Android device.

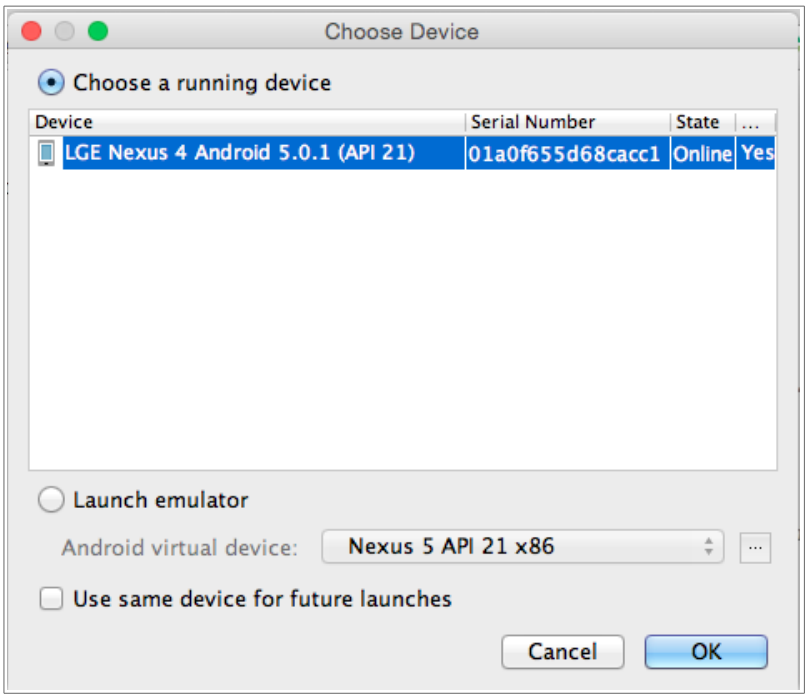


Illustration 3: Choose Device window

6. If your Android project contain error and or asked to sync your gradle file, open **build.gradle(Module: app)** file in **Gradle scripts** directory of the project then click **Sync Now...** button at the top right corner of **Editor** window to sync the project gradle (Illustration 4).

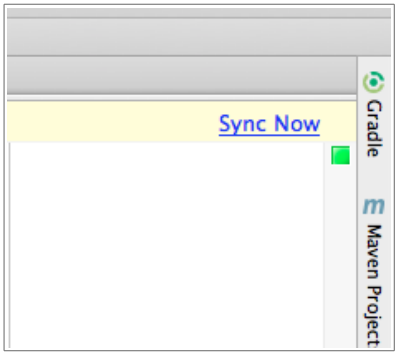


Illustration 4: Sync gradle file

7. When running this app you will find that the app still comes with sample data. You need to insert your own data and configure the app.

Renaming Application Package Name

Renaming application package name is required before publishing your Android app. Make sure

that your package name is unique as this will be checked when you publish your app on the market. Steps to rename the package name of this project are below:

1. On Android Studio, click gear icon at the top right corner of **Project** pane and uncheck **Compact Empty Middle Packages** option (Illustration 5). The package directory will be separated in individual directories.

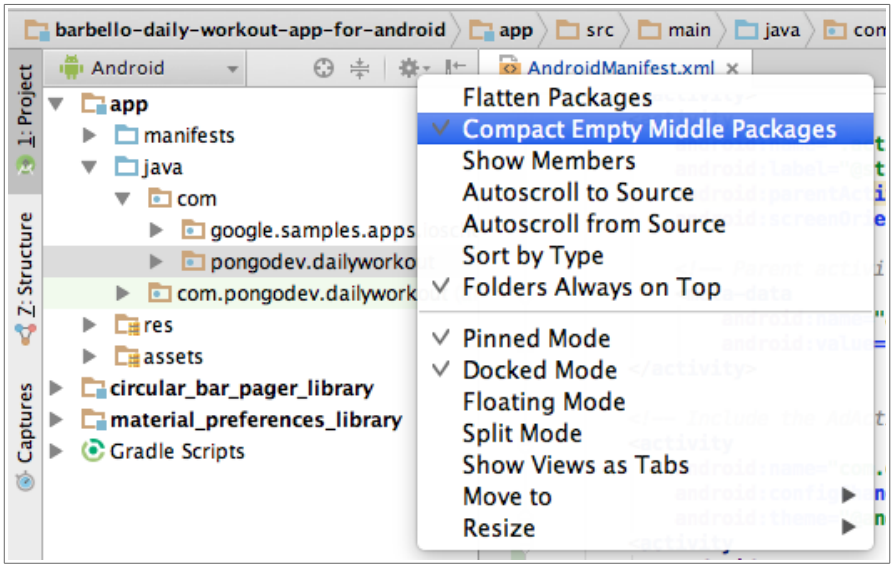


Illustration 5: Uncheck Compact Empty Middle Packages option

2. Select pongodev directory (NOT the one in (androidTest), right click and select **Refactor > Rename**. In the pop-up dialog that appear select **Rename directory** and click **OK** button then warning window will appear. Select **Rename Package** button on that window. **Rename** window will show up, you can rename "**pongodev**" with another name you want. As the package name must be unique, we suggest you to use your website url, For example if you want to create **com.yourwebsite.yourapp** then you need to rename "**pongodev**" with "**yourwebsite**". Click **Refactor** button to process it (Illustration 6).

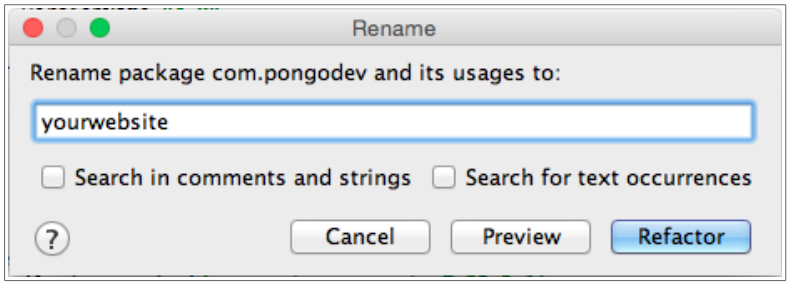


Illustration 6: Rename window

3. Refactoring Preview will appear on **Find** pane at the bottom of Android Studio window.

Click **Do Refactor** (Illustration 7).

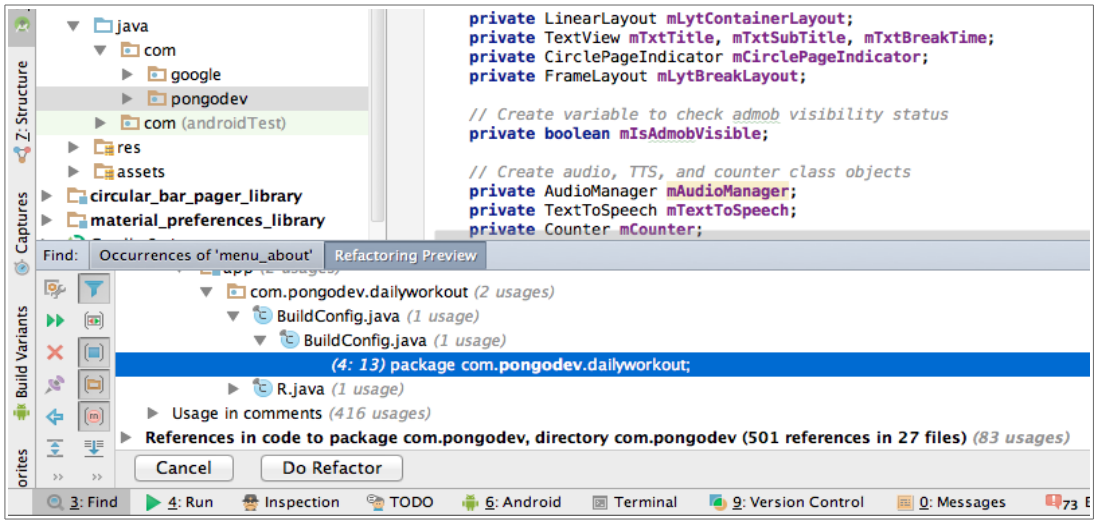


Illustration 7: Refactor Preview on Find pane

4. Expand **yourwebsite** directory and you will find **dailyworkout** directory, rename yourandroidmap directory with another name (for example "**yourapp**") by following the same steps you do previously (Illustration 8).

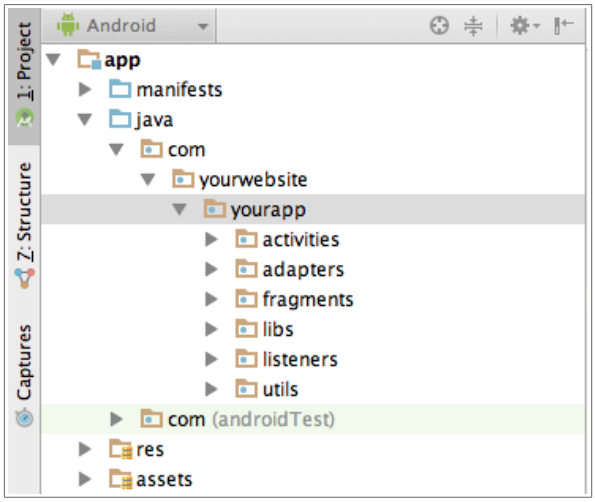


Illustration 8: Package after renamed

5. Open **build.gradle (Module:app)** file in **Gradle Scripts** directory and change **applicationId** with your new package name, for example "**com.yourwebsite.yourapp**" and click **Sync Now** link at the top right corner of the window (Illustration 9).

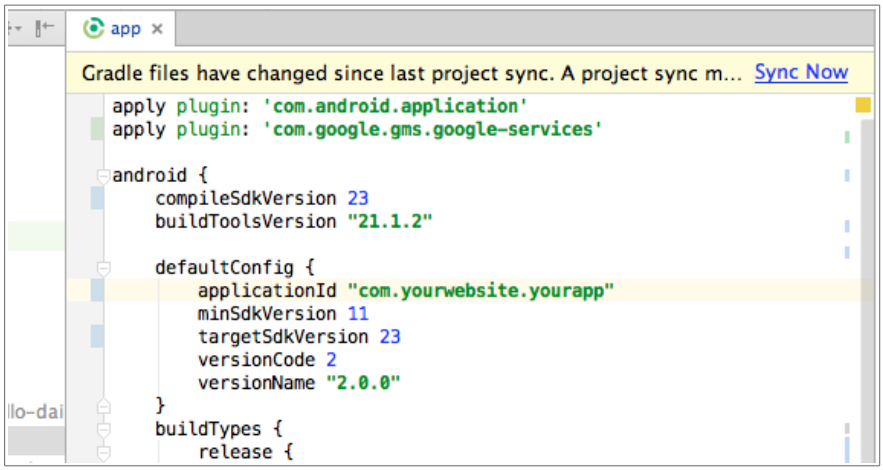


Illustration 9: Change applicationId in build.gradle file

6. Not all files in the project renamed to new package name such as file that is used to handle database path, you need to rename them manually. Find **com.pongodev.dailyworkout** in the following files and renamed them with your new package name:
   - **AndroidManifest.xml** file in **app/manifests** directory. Please check whether package name in all activity tags are changed to the new one if not replace them manually.
   - **Utils.java** file in **app/java/com/utils** directory. The url of sqlite containing package name, change the package name with your package name.

```
ARG_DATABASE_PATH = "/data/data/com.pongodev.yourandroidmap/databases/";
```

7. Try to run the project to see the result. If you follow the steps above correctly, you will see that the app still work properly.

Inserting Data to Database

This app uses SQLite database to stored the workouts data. Below are steps to insert your data into SQLite database:

1. First, download **sqlitebrowser** here and install it on your computer.
2. Run sqlitebrowser and select **File > Open Database…** to open database. Find **db_workouts** file that located in **app/src/main/assets** directory of your Android project by clicking **Open** button.

3. You will see that **db_workouts** consist of three tables, **tbl_categories** table to store categories data, **tbl_workouts** table to store workouts data, and **tbl_images** to store sequence images data of workouts. You need to insert category data first before inserting workouts data as they will be related. Select **Browser Data** tab and select **tbl_categories** under **Table** dropdown. Select **New Record** button to add new category data and **Delete Record** button to delete category data (Illustration 10).

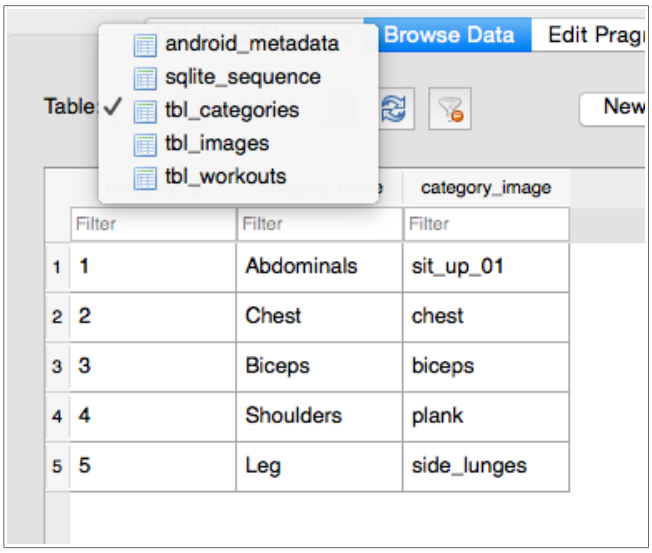


Illustration 10: Add data to tbl_categories

4. By default, there are already categories data sample within the **tbl_categories** table, you can delete them and add your own data or update the data by double click on data under **category_name** and **category_image** field (Illustration 11). As **category_id** is auto increment you do not need to insert it, you only need to insert **category_name** and **category_image** data but remember to take a note of all **category_id** because they will be used in **tbl_workouts** table. For **category_image** field you can use one of images of the workouts under the category. Please insure that the files name are in lower case, without space, and the first character is not number. Image file format must be in **JPG**, then insert the name of category image file in **category_image** field of **tbl_categories** table. Insert the name of the file only, without extension.

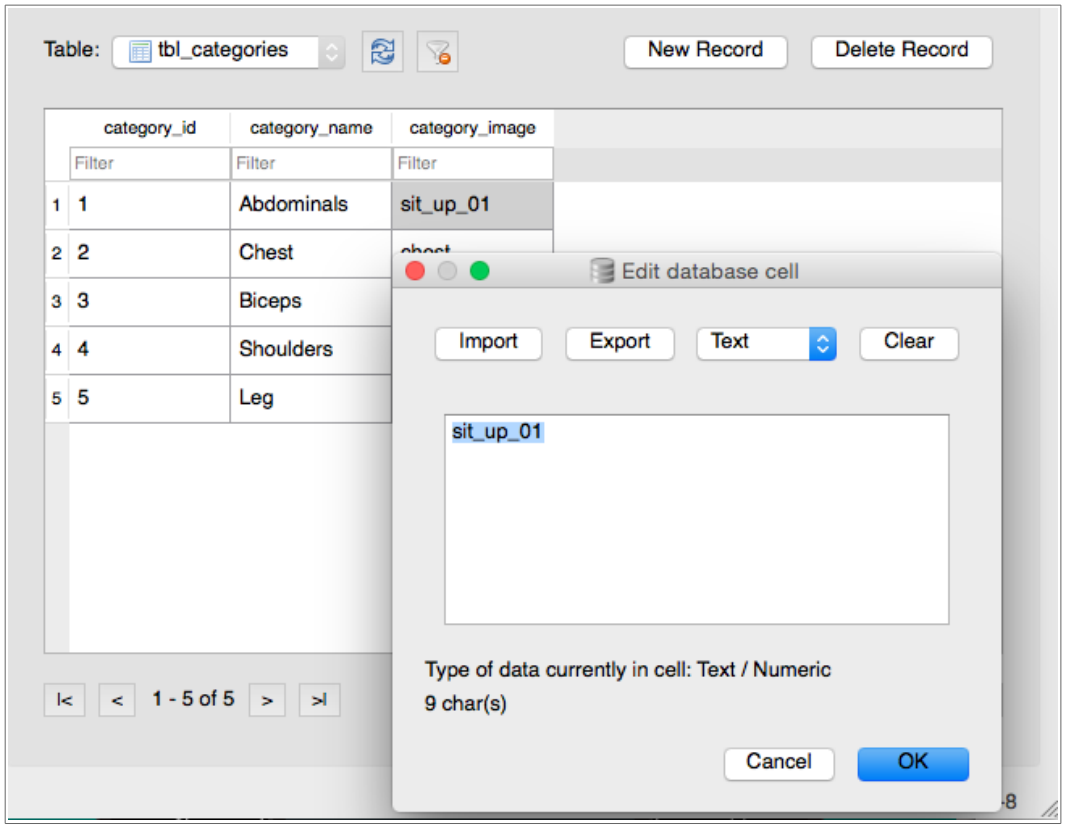
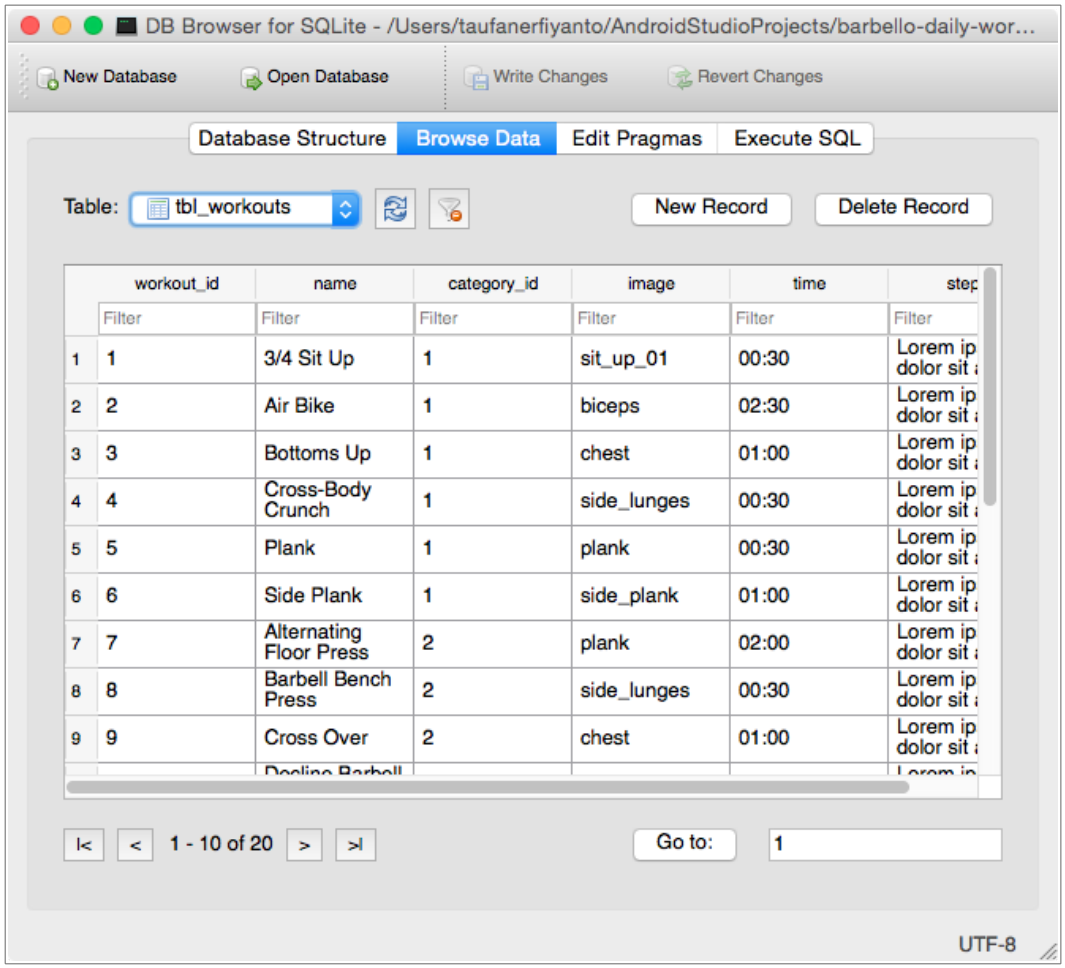


Illustration 11: Update category data

5. After inserting or updating categories data, the next data you need to insert is workouts data. To insert workouts data select **tbl_workouts** in **Table** dropdown (Illustration 12). Please delete all sample workouts data in that table and insert yours by clicking **New Record** button.



	workout_id	name	category_id	image	time	step
1	1	3/4 Sit Up	1	sit_up_01	00:30	Lorem ip dolor sit
2	2	Air Bike	1	biceps	02:30	Lorem ip dolor sit
3	3	Bottoms Up	1	chest	01:00	Lorem ip dolor sit
4	4	Cross-Body Crunch	1	side_lunges	00:30	Lorem ip dolor sit
5	5	Plank	1	plank	00:30	Lorem ip dolor sit
6	6	Side Plank	1	side_plank	01:00	Lorem ip dolor sit
7	7	Alternating Floor Press	2	plank	02:00	Lorem ip dolor sit
8	8	Barbell Bench Press	2	side_lunges	00:30	Lorem ip dolor sit
9	9	Cross Over	2	chest	01:00	Lorem ip dolor sit

Illustration 12: Workouts data in tbl_workouts

6. Insert value in **category_id** field with **category_id** that available in **tbl_categories** table. For example, if in **tbl_categories** table there is category called **Abdominal** with category id **1**, then when you insert workout name "Plank" in **tbl_workouts** table you need to insert 1 (category id of Abdominal) in **category_id** field of **tbl_workouts** table (Illustration 13).

	workout_id	name	category_id	image
1	1	3/4 Sit Up	1	sit_up_01
2	2	Air Bike	1	biceps
3	3	Bottoms Up	1	chest
4	4	Cross-Body Crunch	1	side_lunges
5	5	Plank	1	plank
6	6	Side Plank	1	side_plank

Illustration 13: Insert category id in tbl_workouts table

7. Other fields that you need to insert are workout names, image, time, and steps. Use "MM:SS" format for time data (Illustration 14).

image	time	steps
Filter	Filter	Filter
sit_up_01	00:30	Lorem ipsum dolor sit ame…
biceps	02:30	Lorem ipsum dolor sit ame…
chest	01:00	Lorem ipsum dolor sit ame…
side_lunges	00:30	Lorem ipsum dolor sit ame…

Illustration 14: Insert time and image data

8. For image field you need to store all of your workout image files in **res/drawable** directory of your Android project, you can delete the sample of workout images in **drawable** directory. Please insure that the files name are in lower case, without space, and the first character is not number. Image file format must be in **JPG** and maximum size is **500 x 500 pixels**, then insert the name of image files in **image** field of **tbl_workouts** table. Insert the name of the file only, without extension (Illustration 15).

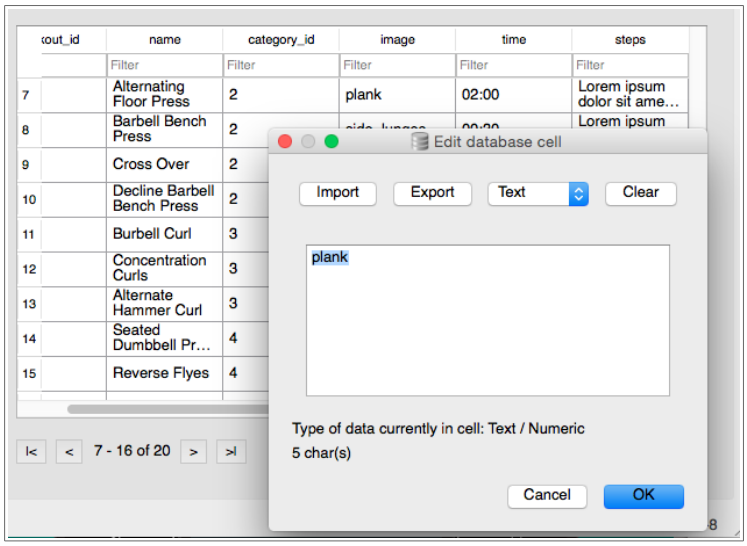


Illustration 15: Insert image name in image field of tbl_workouts table

9. If you have sequence images for workout data you can insert them in **tbl_images** table. This images will be used for animated image. To insert the images, under **tbl_images** table delete all the sample data first by selecting all data and click **Delete Record** button. Click **New Record** button to insert new image data. Insert workout id under **workout_id** field and workout image under **image** field (Illustration 16).

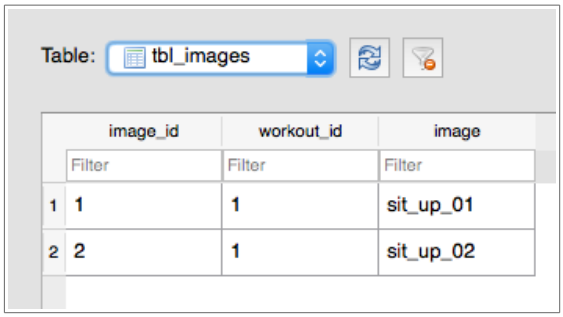


Illustration 16: Insert sequence images data in tbl_images table

10. If you have already inserted all of your workouts data, do not forget to save your database. Click **Write Changes** button.

11. Run the app to see the result.

Setting Up Admob

To set up Admob make sure that you have already upgraded your Admob account. Go to apps.admob.com and login with your Admob account, if you have not upgraded your account yet please see this guide from Google about how to do it. Below are steps to set up Admob on Android project:

1. Login to Admob with your account and select **Monetize** menu, on Monetize window click **Monetize new app** button.

2. Select **Add your app manually** and fill **App name** with your application name and **Platform** with Android, click **Add app** button when done (Illustration 17).

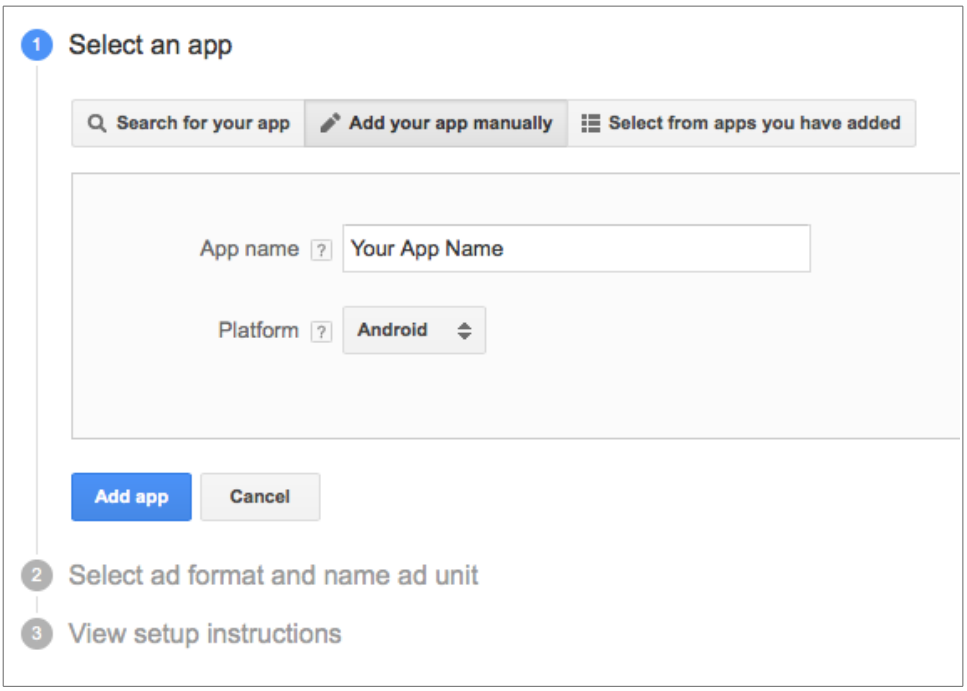


Illustration 17: Add app manually

3. Next step is selecting the ad type, select **Banner** and do some configuration to the banner, click **Save** button when finish (Illustration 18).

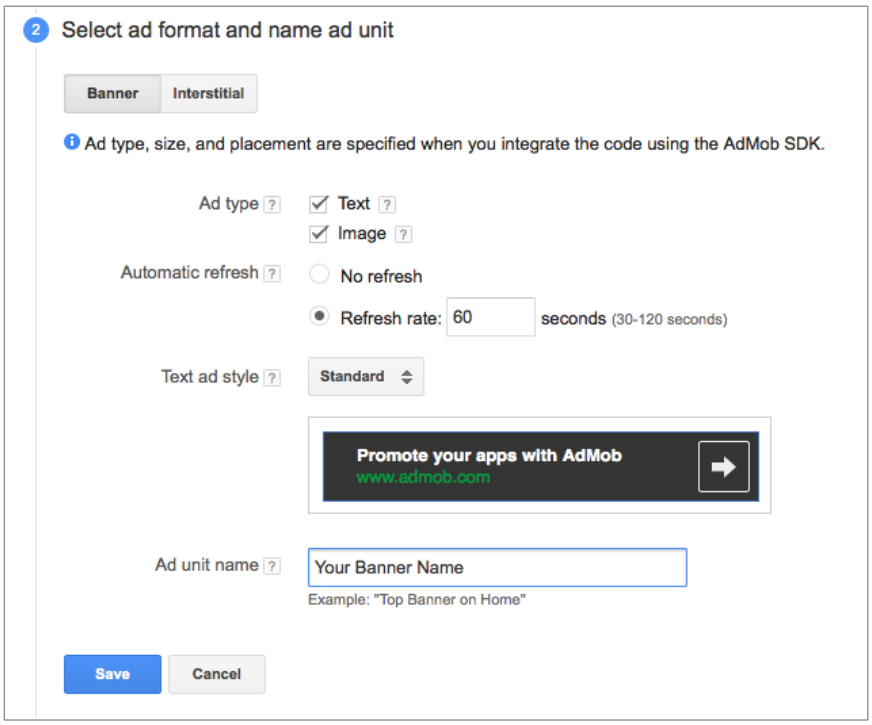


Illustration 18: Select ad format and ad configuration

4. You will get the **Ad unit ID** (Illustration 19), copy that ID and click **Done** button to finish.

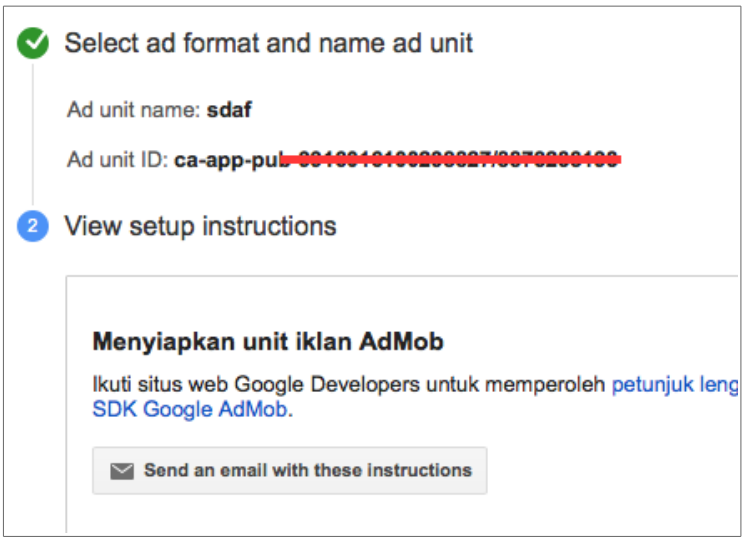


Illustration 19: Ad unit ID

5. Open **strings.xml** file in **app/res/values** directory and find **BANNER_AD_UNIT_ID** and replace it with your Ad unit ID.

6. As this app use two ad formats, banner and interstitial ad, you need to create one ad unit for interstitial. Click **Monetize** and select your app name on the left side and select **+ New ad unit** button to create new ad unit (Illustration 20).

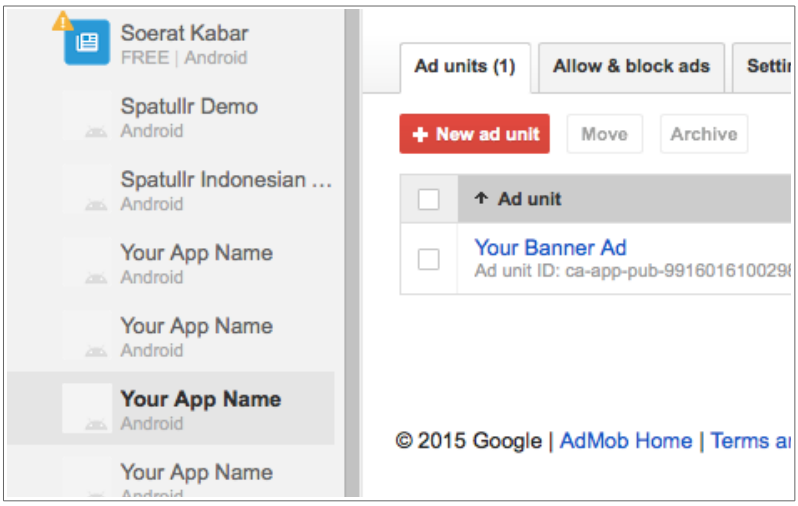


Illustration 20: Create new ad unit

7. In ad format option, select **Interstitial**. And the rest configuration is similar with ad banner format. Click **Save** button when finish (Illustration 21).

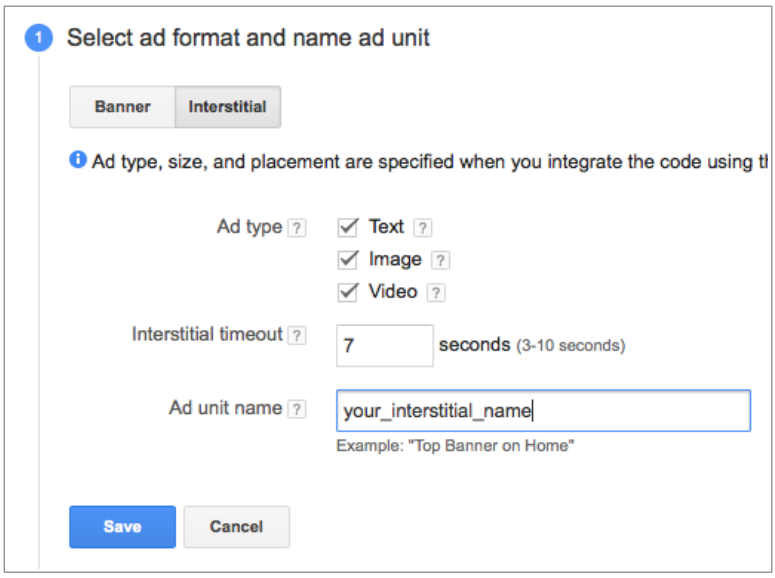


Illustration 21: Configure interstitial ad

8. You will get ad unit id for interstitial ad, copy the ID. In **strings.xml** find **INTERSTITIAL_AD_UNIT_ID** and replace it with the your interstitial ad unit ID.

9. If you have already published your app on Google Play you can link your app with Admob by clicking **Link your app** link on applications list on Admob **Home** page. Search your app name on **Search** box and click **+ Select** button to link the app (Illustration 22). You can skip this step if you have not published your app on Google Play yet.

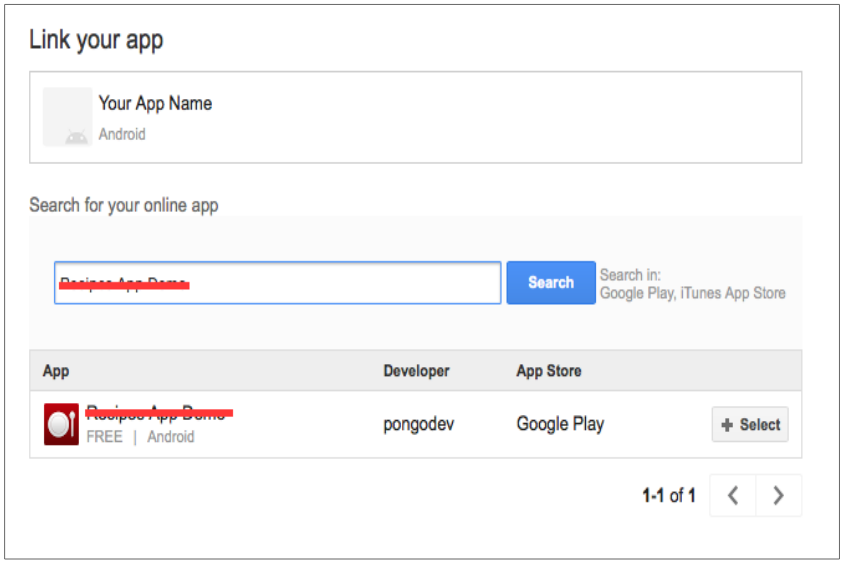


Illustration 22: Link app with Google Play

10. Other ad configurations that also important are located in **Utils.java** files. Open **Utils.java** files in **app/java/com/yourwebsite/yourapp/utils** directory of your Android project. Find

the following code,

```
// For every ActivityDetail display you want to interstitial ad show up.
// 3 means interstitial ad will show up after user open ActivityDetail
page three times.
public static final int ARG_TRIGGER_VALUE = 3;
// Admob visibility parameter. Set true to show admob and false to hide.
public static final boolean IS_ADMOB_VISIBLE = false;
// Set value to true if you are still in development process, and false
if you are ready to publish the app.
public static final boolean IS_ADMOB_IN_DEBUG = false;
```

You can configure when the interstitial ad will be display by changing value of **ARG_TRIGGER_VALUE**, 3 means interstitial ad will be displayed every three times users open detail screen. For ad visibility you can change value of **IS_ADMOB_VISIBLE** to **true** if you want to use Admob in your application and **false** to hide it. During the development process you need to set value of **IS_ADMOB_IN_DEBUG** to **true** and when you want to release it you need to set this value to **false**.

11. You can run the app to see the result of this ad configuration.

Customizing Application Color

Barbello has implemented material design in its user interface so that customizing the color of the app become easier. Here is how to change the color of the app:

1. Open **colors.xml** in **app/res/values** directory.
2. Change the hex color of each color attributes (Illustration 23).

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <!-- app main color (toolbar color) configuration. You can change the app color here. -->
    <color name="primary_color">#009688</color>
    <color name="dark_primary_color">#00796B</color>
    <color name="light_primary_color">#B2DFDB</color>
    <color name="text_and_icon_color">#FFFFFF</color>
    <color name="accent_color">#CDDC39</color>
    <color name="primary_text">#212121</color>
    <color name="secondary_text">#727272</color>
    <color name="divider_color">#455A64</color>
    <color name="window_background_color">#263238</color>
    <color name="material_background_color">#37474F</color>
</resources>
```

Illustration 23: Customize app color

3. Illustration 24 show you what each color attributes used for in the application user

interface.

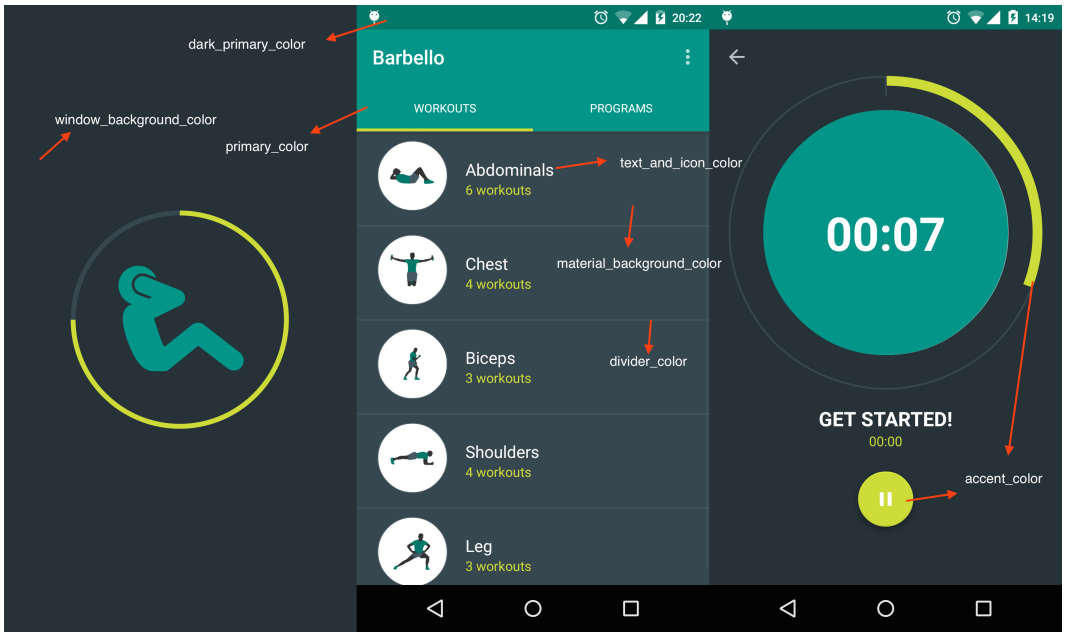


Illustration 24: Use of color attributes in application user interface

4. If you want to change the color based on Google Material Design guidelines, you can use tools from materialpalette.com as reference.

Customizing Image Resources

To make this app meet your brand, besides customizing the color of the app, you also need to change the image resources which used in this app such as app launcher icon and splash screen logo. Please, follow the steps below to customize the image resources:

1. To change **app launcher icon** of the app, create your own icon in **PNG** format and name it as **ic_launcher.png**. Create it in 4 different sizes with the following specifications:
   - 72 x 72 pixels in **res/mipmap-hdpi** directory.
   - 48 x 48 pixels in **res/mipmap-mdpi** director.
   - 96 x 96 pixels in **res/mipmap-xhdpi** directory.
   - 192 x 192 pixels in **res/mipmap-xxhdpi** directory.
2. Put those files in each directories via window explorer.

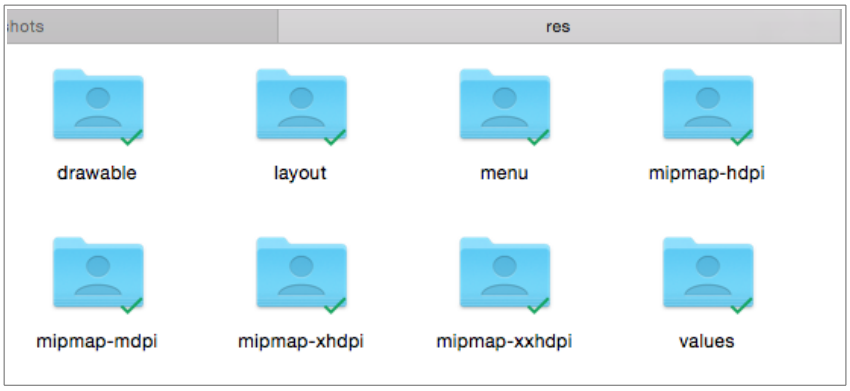


Illustration 25: Access mipmap directories via window explorer

3. Next is splash screen logo, create image in **PNG** format and name it as **splash_screen_logo.png**. Create it in 4 different sizes with the following specifications:
    - 450 x 450 pixels in **res/mipmap-hdpi** directory.
    - 300 x 300 pixels in **res/mipmap-mdpi** directory.
    - 600 x 600 pixels in **res/mipmap-xhdpi** directory.
    - 900 x 900 pixels in **res/mipmap-xxhdpi** directory.

Customizing App Content

The last customization is app content, you can customize the app content such as application name, Google Play url, and other content via **strings.xml** that located in **app/res/values** directory. For Google Play url you need to change **com.your.package** with your own package name.

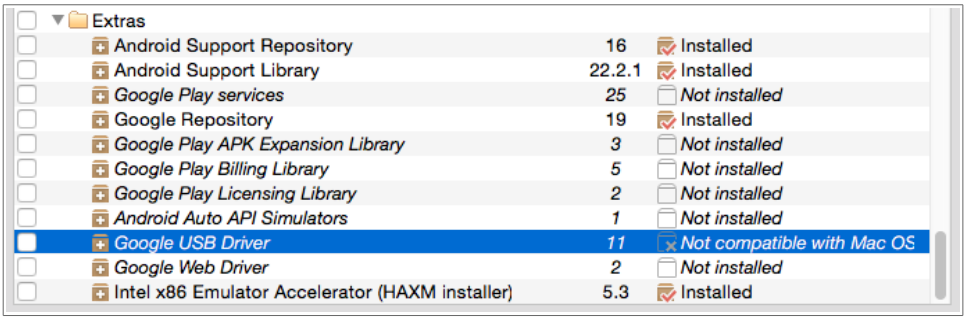
```
    </string-array>

    <!-- DO NOT change preferences key below -->
    <string name="pref_share_key">prefShareKey</string>
    <string name="pref_rate_review_key">prefRateReviewKey</string>

    <!-- preferences value of about preferences page -->
    <string name="pref_app_title">Application Name</string>
    <string name="pref_app_summary">Barbello</string>
    <string name="pref_version_title">Build Version</string>
    <string name="pref_version_summary">Version 2.0</string>
    <string name="pref_developer_title">Developer</string>
    <string name="pref_developer_summary">pongodev</string>
    <string name="pref_copyright_title">Copyright</string>
    <string name="pref_copyright_summary">Copyright 2014 pongodev. All rights reserved</string>
    <string name="pref_share_title">Tell a Friend</string>
    <string name="pref_share_summary">Tell your friend about this app</string>
    <string name="pref_rate_review_title">Rate &amp; Review</string>
    <string name="pref_rate_review_summary">Rate and review this app if you think it is awesome</string>

    <!-- Google Play url. Change your.package.name with your own package name. -->
    <string name="google_play_url">https://play.google.com/store/apps/details?id=your.package.name</string>

    <!-- Banner ad id. Insert your own banner id, you can get banner id via admob. -->
    <string name="banner_ad_unit_id">BANNER_AD_UNIT_ID</string>

    <!-- Interstitial ad id. Insert your own interstitial id, you can get interstitial id via admob. -->
    <string name="interstitial_ad_unit_id">INTERSTITIAL_AD_UNIT_ID</string>

    <!-- Sharing content -->
    <string name="share_to">Share to</string>
    <string name="subject">Stay Fit with This App</string>
    <string name="message"> is fitness app that will help you stay fit everyday. Download it at </string>
    <string name="sent_via_message">This message was sent from</string>
    <string name="download">Download</string>
    <string name="at">at</string>
    <string name="share_program_message">Hi, I just completed my</string>
```

Illustration 26: Customize app content in strings.xml

Running the Application

To run Android project on Android device, please follow the steps below:

1. First, open **SDK Manager** by clicking **Tools > Android > SDK Manager**. Under **Extras** insure that **Google USB Driver** has been installed, if not, download and install the package (note that Google USB Driver is not compatible in Mac OSX so you do not need to install it if you are using Mac OSX) (Illustration 27).



Extras		
Android Support Repository	16	Installed
Android Support Library	22.2.1	Installed
Google Play services	25	Not installed
Google Repository	19	Installed
Google Play APK Expansion Library	3	Not installed
Google Play Billing Library	5	Not installed
Google Play Licensing Library	2	Not installed
Android Auto API Simulators	1	Not installed
Google USB Driver	11	Not compatible with Mac OS
Google Web Driver	2	Not installed
Intel x86 Emulator Accelerator (HAXM installer)	5.3	Installed

Illustration 27: Install Google USB Driver

2. Enable **USB Debugging** on your device, you can read here about how to enable USB debugging on android device.

3. Connect your Android device to your computer with USB cable.

4. After that, on Android Studio select **Run > Run 'app'**. Select your device in **Choose Device** window and click **OK** button (Illustration 28).

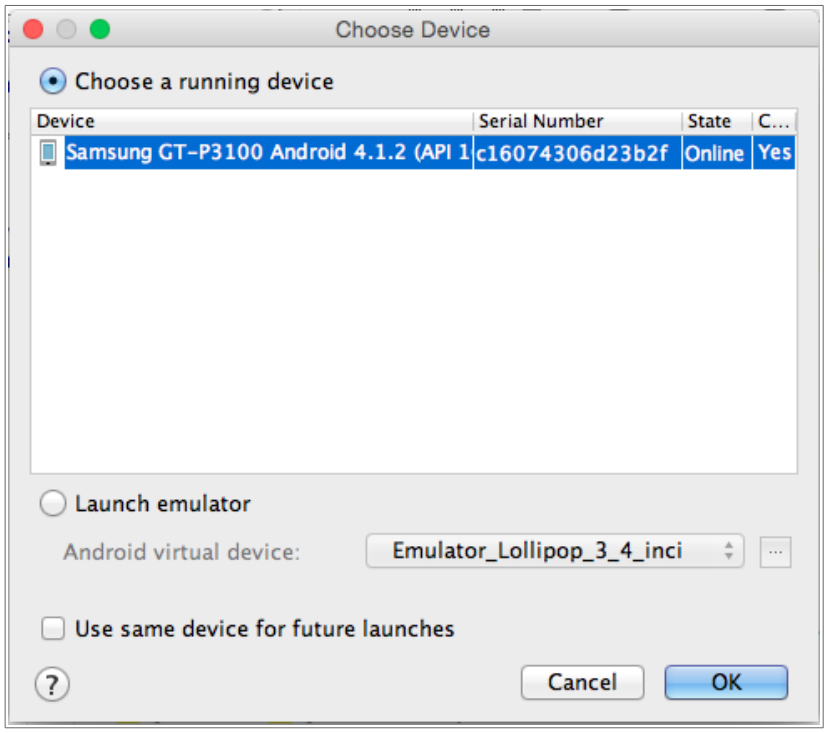


Illustration 28: Select your device on Choose Device window

5. If you can see your device on **Choose Device** window, it means that driver of your Android device is not installed on your computer yet. You need to install your device driver first.

Publishing Android App

After configuring the package name, inserting data, customizing the user interface color image resources, content and ensuring that the app has been run properly, the last step is publishing your app to APK file. Below are step by step to publishing Android project to APK file:

1. on Android Studio, select **Build > Generate Signed APK...**.

2. **Generate Signed APK** wizard window will appear. Select **app** in module section then click **Next** button.

3. In the next step, if you have already created key store file click **Choose existing...** and insert your key store password and alias. If have not created it yet, click **Create new...** to create new key store file.

4. On **New Key Store** window fill all forms that required and click **OK** button (Illustration 29).

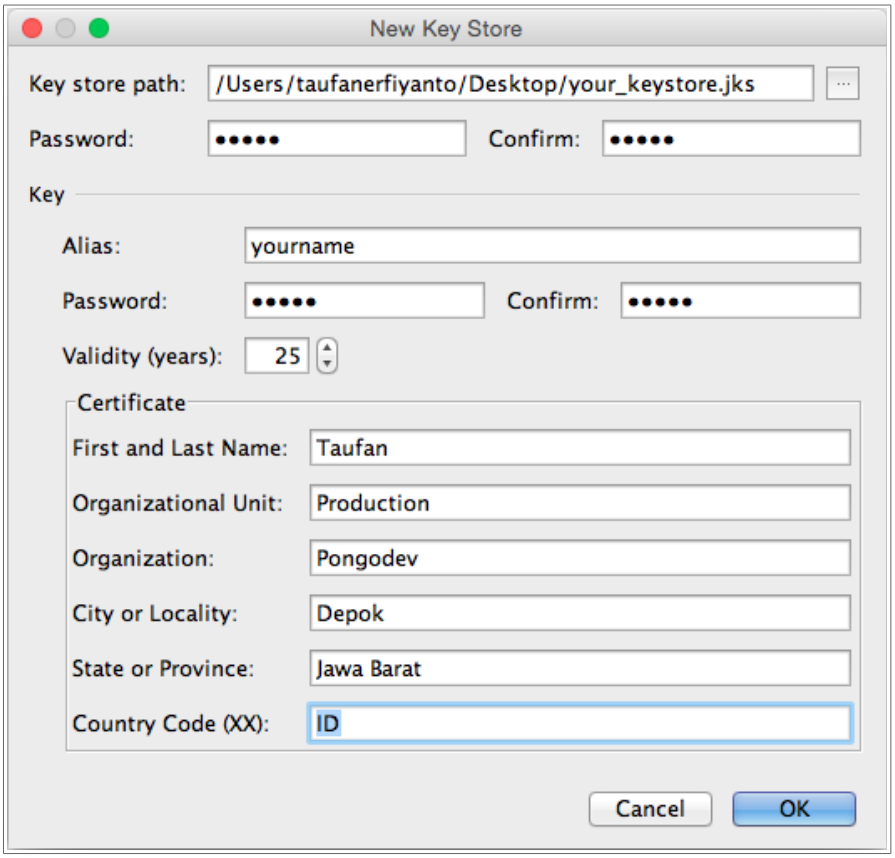


Illustration 29: New Key Store window

5. Last is set the destination folder of APK files, and Build Type to **release** and click **Finish** button. Your APK will be generated, once it finish, go to destination folder to see the file.

Updating App Version

When you want to update your app on Google Play, for example add new workout data or fix minor bugs, make sure that you change the **versionCode** and **versionName** on **build.gradle(Module: app)** file that located in **app/Gradle Scripts** directory to higher number than previous version and click **Sync Now** link (Illustration 30). Re-generate APK again using the same keystore that you use for first version of the app.

```
android {
    compileSdkVersion 23
    buildToolsVersion "21.1.2"

    defaultConfig {
        applicationId "com.pongodev.dailyworkout"
        minSdkVersion 11
        targetSdkVersion 23
        versionCode 2
        versionName "2.0.0"
    }
```

Illustration 30: Change versionCode and versionName to update the app

About the Author

Hello, we are pongodev, mobile developer team from Depok, West Java, Indonesia. founded on 2012 we started building and selling mobile applications as a template on Marketplace. our apps usually focusing on ease of use and good design.

Website: http://pongodev.com

Contact: contact@pongodev.com

Facebook: https://www.facebook.com/pongodev

Twitter: https://twitter.com/pongodev