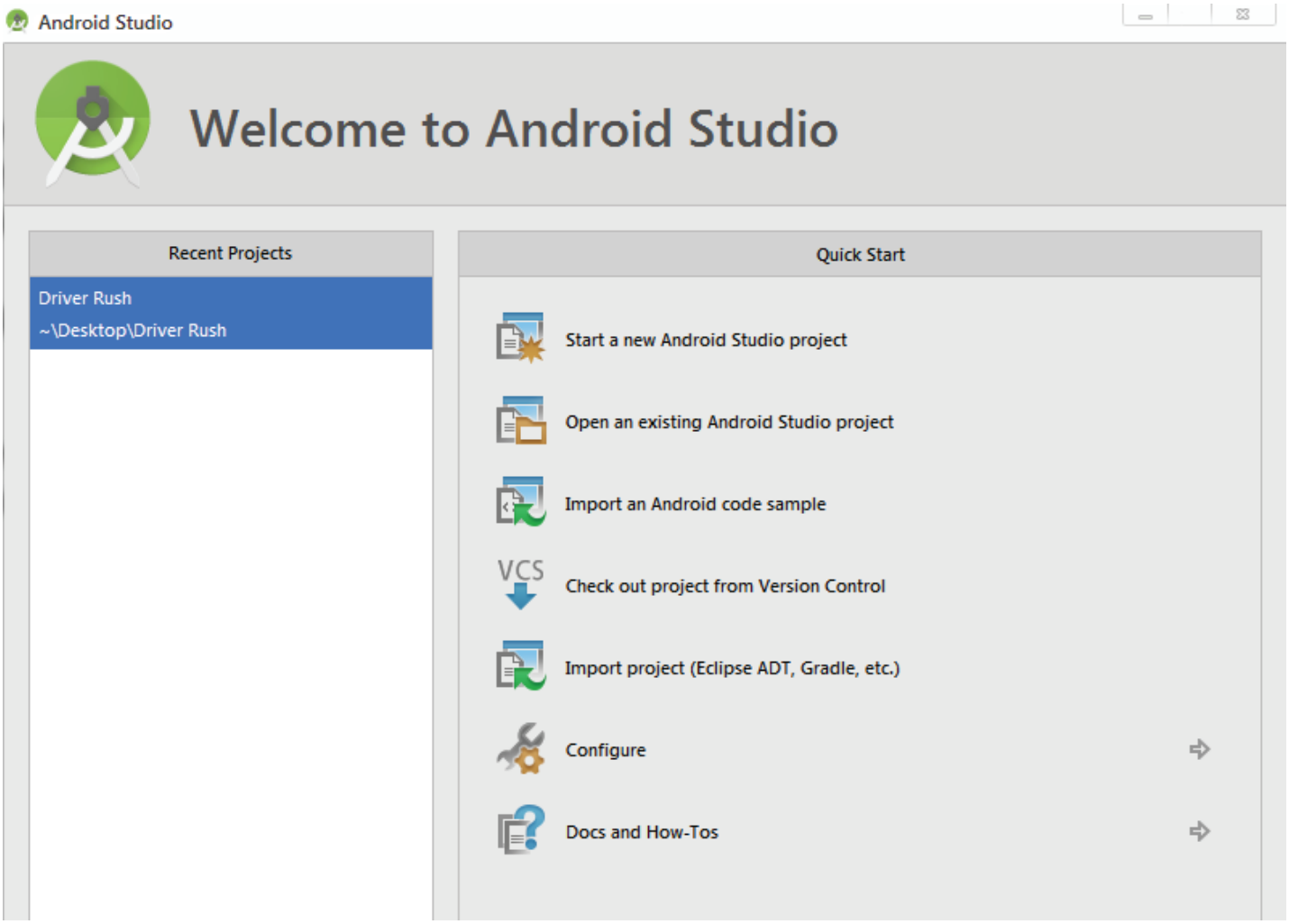
DRIVER RUSH DOCUMENTATION

Step 1: Importing Driver Rush

Driver Rush was programmed in Android Studio. It is **recommend that you use Android Studio** to modify the source for Driver Rush and this manual will work on the assumption that you are using Android Studio. If you haven't already, you should go install it now.

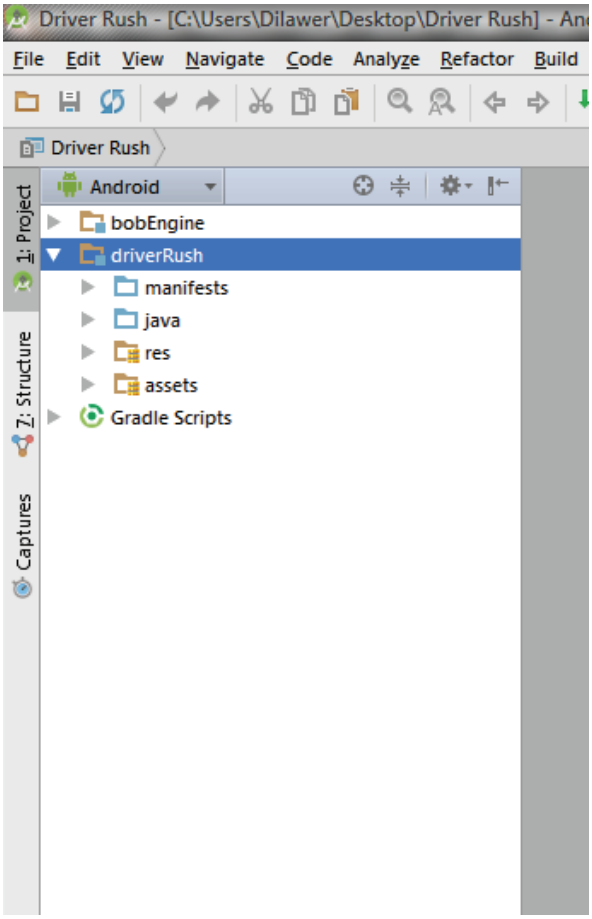
Downloads can be found at: http://developer.android.com/sdk/index.html

1. Extract the Driver Rush folder from Files.zip.
2. From the Android Studio welcome screen (shown below) choose "Open an existing Android Studio project"



Navigate to the location where you extracted the Driver Rush folder and select it. Click "Okay". Wait for Android Studio to load up all the code and resources.

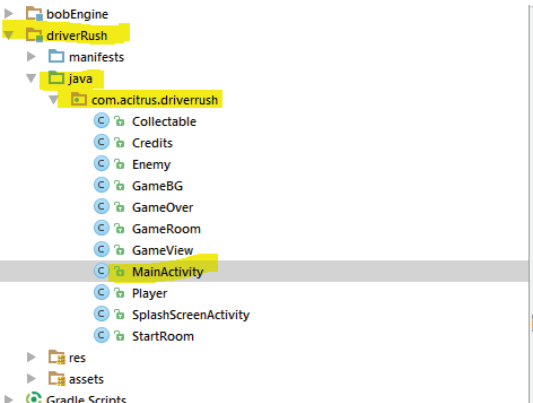
*From this point you will have your project viewer like this.



You're only going to need to work with driverRush folder as that is where all your resources and files are for the main game. **Ignore the bobEngine folder, that is used to develop the game.**

Step 2 : Adding your AdMob Banner and Interstitial IDs

In the driverRush project open "driverRush > java > com.acitrus.driverrush > MainActivity.java". Near the top is a data dictionary with "Constants", "AdMob", and "Music" headings. Under "AdMob", find BANNER_AD_UNIT_ID and INTER_AD_UNIT_ID. Replace their values with your banner ad unit ID and interstitial ad unit ID, respectively. Run the application, and the ads should display properly.




```
package com.acitrus.driverrush;

import ...

/**
 * Crazy Taxi Driver Android game Bobby Lou Jo (2014)
 *
 * Crazy Taxi Driver is a game created by Bobby Lou Jo. It is available as a
 * template on Chupamobile. Those who have purchased a license to use this
 * source from Chupamobile may reskin and make other changes to the game then
 * use it for commercial or non-commercial purposes in accordance to the
 * licensing terms on Chupamobile.
 *
 * @author Benjamin Blaszczak (Bobby Lou Jo)
 *
 */
public class MainActivity extends acActivity {

    // AdMob
    private static final String BANNER_AD_UNIT_ID = "YOUR BANNER AD ID";          // "Top banner on Game Over" ad un
    private static final String INTER_AD_UNIT_ID = "YOUR INTERSTITIAL AD ID";     // "Interstitial ad" ad unit ID

    public ProgressBar spinner;                                 // Loading circle to show while loading interstitial ad
    private AdRequest adRequest;                                // AdMob ad request
    private AdView adView;                                      // The banner
    private InterstitialAd inter;                               // Interstitial (full screen) ad.
    private boolean show;                                       // Should the interstitial be show when it is loaded?

    // Music
    private MediaPlayer backgroundMusic;       // MediaPlayer for playing background music

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

To change the frequency of the interstitial ads: open GameOver.java and replace the value of AD_FREQ with the number of plays you would like between interstitial ad displays.

Step 3: Replacing the Graphics

Replacing the graphics in Driver Rush is very straight forward. Simply created graphics that mimic the graphics in the res/drawable folder. Make sure your graphics have the same file names and then replace the graphics in the drawable folder with yours.

Some notes:
- Your graphics do not have to be the same resolutions as the default graphics. They should, however, be the same aspect ratio. For example: the default title.png is 64x32 pixels. So, your title.png should also be twice as wide as it is tall (ex. 400x200)

- Some graphics have multiple frames (enemy.png, score.png, and numbers.png). Each frame has the same dimensions and they are stacked vertically. Your graphics should include the same number of frames in the same configuration (again, with the same aspect ratio).

- companylogo.png should be replaced with your company logo.

Step 4: Adjusting the Difficulty and Gameplay

The difficulty and other game play factors can be easily adjusted. GameRoom.java, Enemy.java, Collectable.java, and Player.java all contain constants that determine how the game behaves. Here are some descriptions of how each constant affects game play:

GameRoom.java

INIT_SPEED: The initial speed of the player. Making this bigger will make the player faster from the start of the game.

Driver Rush

DEC: This constant inversely affects the magnitude of the deceleration of the player. If you make this constant bigger, it will take longer for the player to slow down.

ACC: This constant affects the acceleration of the player after picking up a collectable. The bigger this is, the quicker the player will get up to speed after getting a collectable.

SPEED_THRES: This is the speed at which the collectables start to have less of an effect on the player's speed. When the player has reached this speed the speed increase from collectables will be inversely proportional to the player's speed. This is kind of like a flexible max speed.

NUM_ENEMIES: The number of enemy objects to create. This does not change how frequently they appear. However, if you do change the frequency that they appear (with RELEASE) you may need to add more or take some away.

RELEASE: The vertical position on the screen that the previous enemy has to pass before the next one is sent. This is as a fraction of the screen, where 0 is the bottom and 1 is the top. So, 0.5 would be the middle of the screen. 0.75 would be closer to the top. The closer this value is to 1, the more frequently enemies will be released.

Enemy.java

INIT_LC_SPEED: The speed at which enemies move from lane to lane.

SWITCH_THRES: The vertical position on the screen at which the enemies can no longer change lanes. This value follows the same rules as RELEASE in GameRoom.java.

Player.java

INIT_LC_SPEED: The speed at which the player moves from lane to lane.

Collectable.java

SPEED_UP: The speed increase that the player gets when they pick up a collectable.

Thanks for the purchase. Goodluck!