MODERN WORD SEARCH GAME DOCUMENTATION

Warning

Please read this document fully, it doesn't only explain how to configure the app but also how to troubleshoot problems in case you run into them.

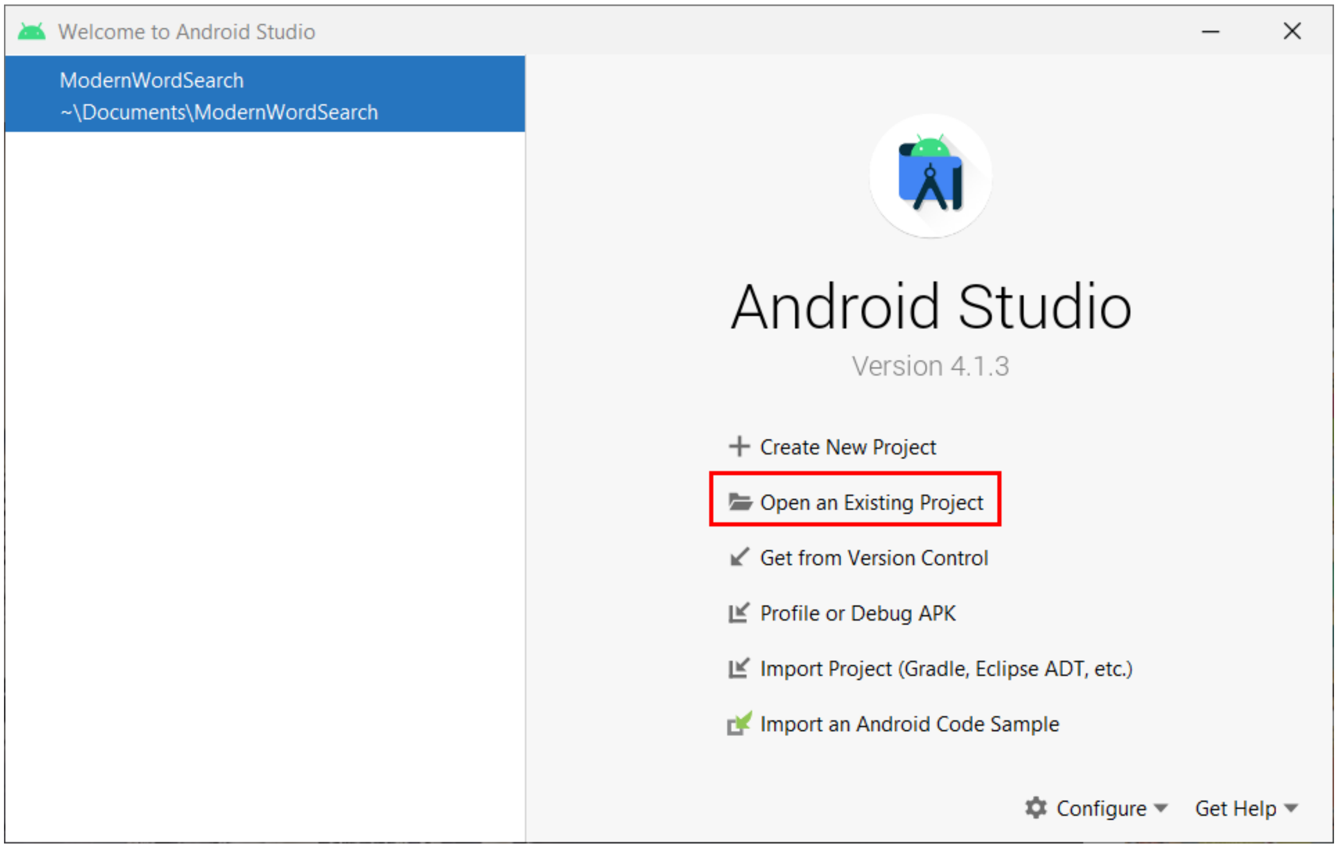
Ads will not show up until you configure your GDPR settings. Please read this document to see how to configure it.

Thanks for purchasing the Modern Word Search Game. Please note that you must make some changes in the game otherwise Google may remove your app and account from Play Store due to repetitive content. Your game should look and feel different to other people's game who have purchased the game previously. Therefore, if you don't make any changes you risk both your self and others. The details of such modifications will be listed in the following sections.

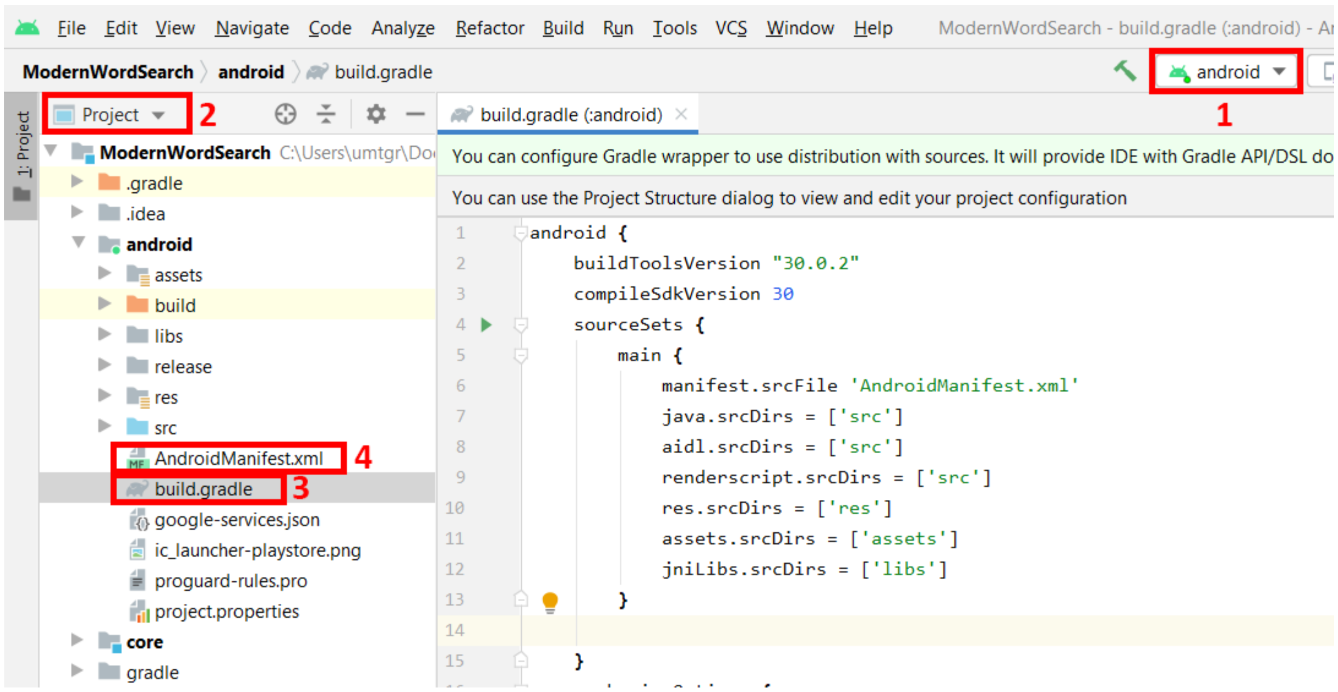
Setting up the Project

1. Unzip the project you downloaded from Codecanyon.

2. Open Android Studio. Close any existing projects and open the project named ModernWordSearch in the unzipped folder:



3. Wait for a while until the sync process ends. Make sure Android (1) and Project (2) are selected as shown below and open build.gradle (3), then AndroidManifest.xml (4):

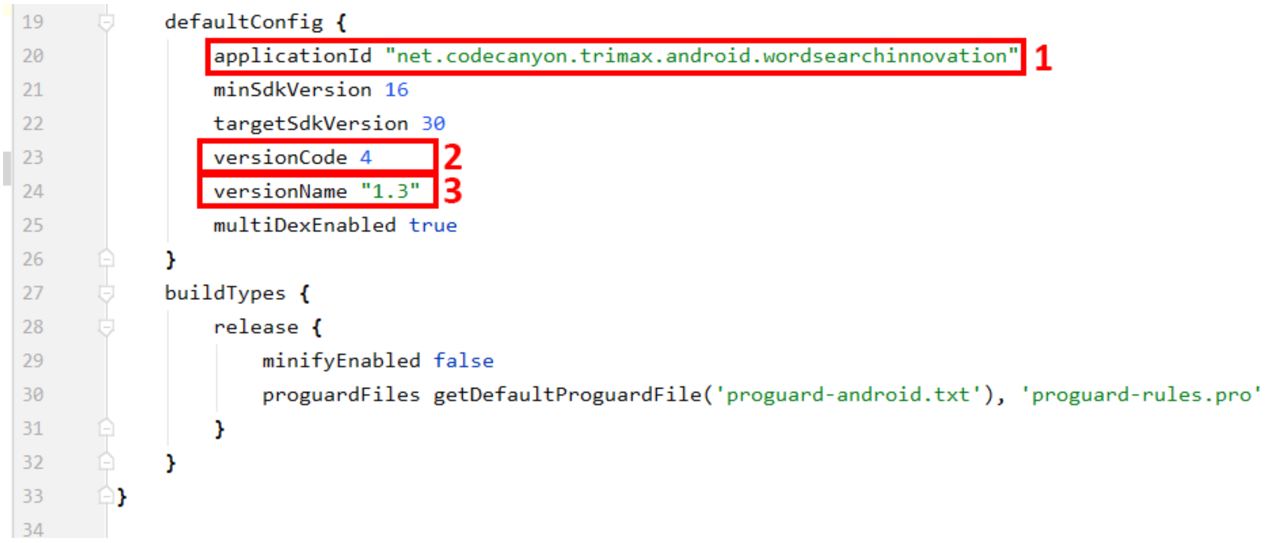


You can configure Gradle wrapper to use distribution with sources. It will provide IDE with Gradle API/DSL do

You can use the Project Structure dialog to view and edit your project configuration

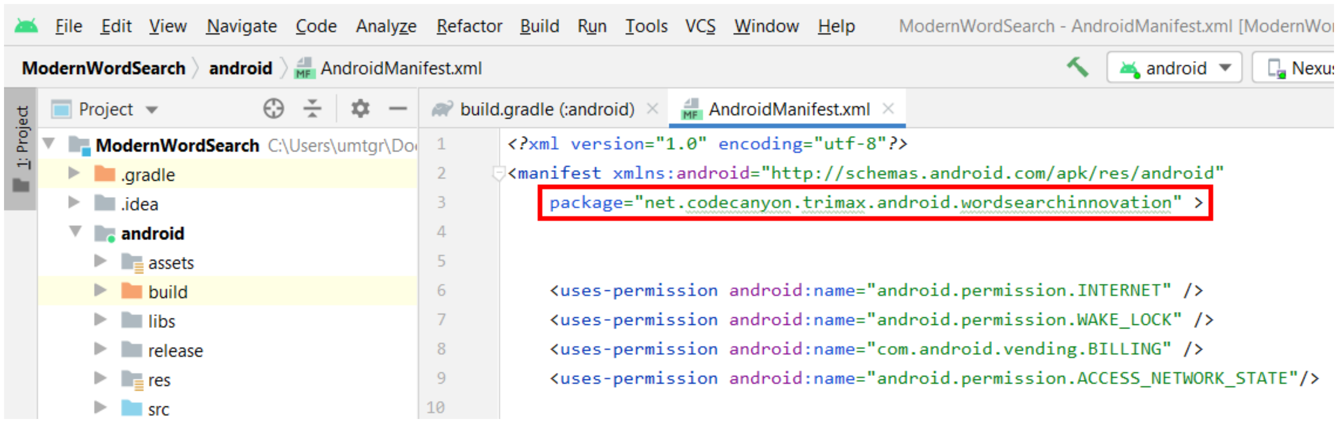
```
android {
    buildToolsVersion "30.0.2"
    compileSdkVersion 30
    sourceSets {
        main {
            manifest.srcFile 'AndroidManifest.xml'
            java.srcDirs = ['src']
            aidl.srcDirs = ['src']
            renderscript.srcDirs = ['src']
            res.srcDirs = ['res']
            assets.srcDirs = ['assets']
            jniLibs.srcDirs = ['libs']
        }

    }
```

4. In In build.gradle file, change the **applicationId (1)**, **versionCode (2)** and **versionName (3)**. applicationId uniquely identifies your app at Play Store, it has nothing to do with the name of game. You can replace the word trimax with your Codecanyon username and it will be unique. Make versionCode 1 and versionName 1.0.0:



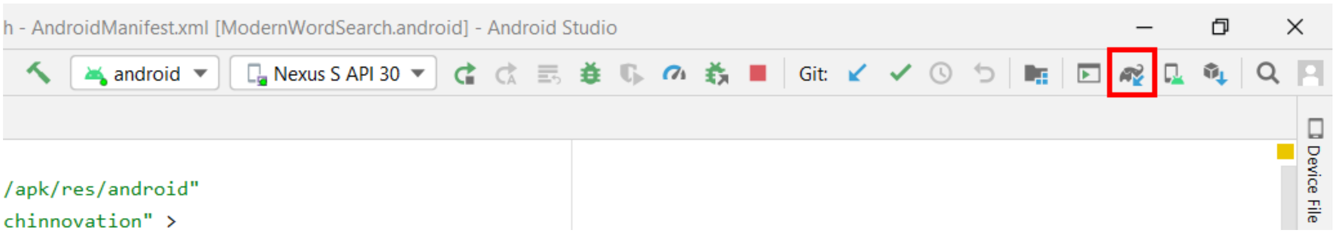
```
    defaultConfig {
        applicationId "net.codecanyon.trimax.android.wordsearchinnovation"
        minSdkVersion 16
        targetSdkVersion 30
        versionCode 4
        versionName "1.3"
        multiDexEnabled true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
```

5. In AndroidManifest.xml you will enter the same applicationId you entered in the previous step into the package part:


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
ModernWordSearch - AndroidManifest.xml [ModernWo
ModernWordSearch
android
AndroidManifest.xml
android
Nexu
Project
build.gradle (:android)
AndroidManifest.xml
ModernWordSearch C:\Users\umtgr\Do
.gradle
.idea
android
assets
build
libs
release
res
src

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="net.codecanyon.trimax.android.wordsearchinnovation" >


    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="com.android.vending.BILLING" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

6. Android Studio will prompt you to sync the project by popping a yellow balloon on the right-top corner of the screen. If it doesn't you should manually sync it by clicking the button shown below:


h - AndroidManifest.xml [ModernWordSearch.android] - Android Studio
android
Nexus S API 30
Git:
Device File

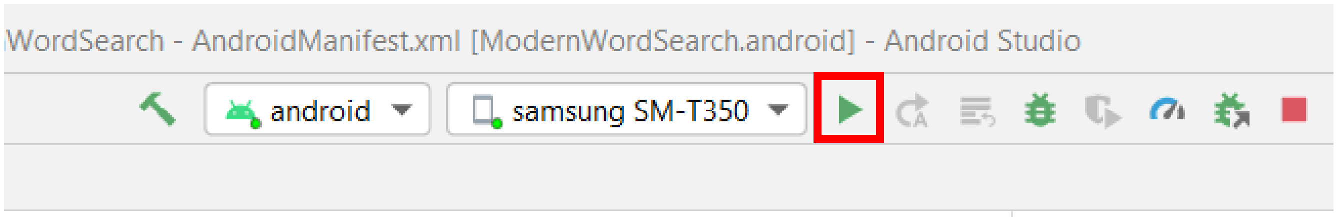
```
/apk/res/android"
chinnovation" >
```

If you get the following error:

Execution failed for task ':android:processDebugGoogleServices'.
> File google-services.json is missing. The Google Services Plugin cannot function without it.

Android Studio gives this error because you don't have a google-services.json file in your android folder. To fix it, you must create a firebase project at: https://console.firebase.google.com/ After you successfully create your project, you should copy and paste the resulting google-services.json file into the android folder. Now sync the project again.

7. Now connect a smartphone or a tablet to your computer and run the app by clicking the green button:


WordSearch - AndroidManifest.xml [ModernWordSearch.android] - Android Studio
android
samsung SM-T350

Configuring the Game

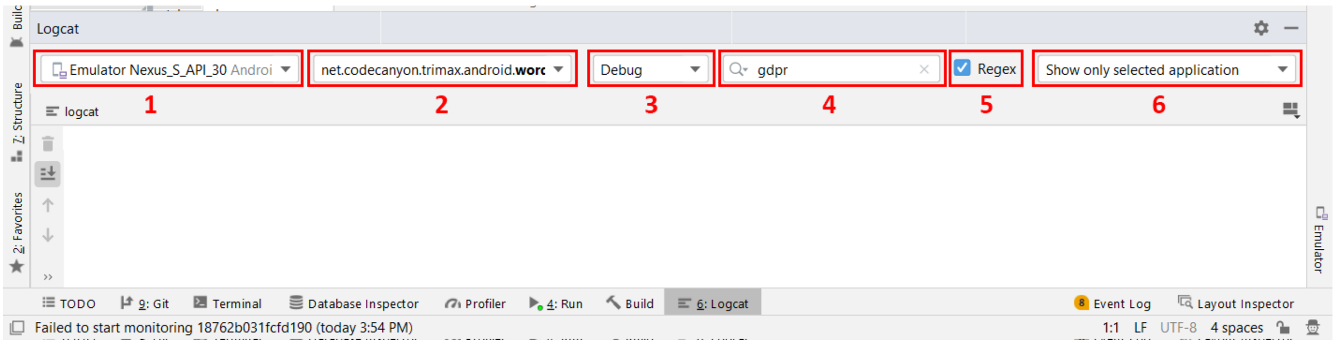
There is a wide variety of configuration options in the project. Most of the options are commented in source files.

1. res/values/strings.xml. This file mainly contains the settings related to Android such as game title, Admob, IAP and game menu items' visibility. Choose a different title for your app other than "Modern Word Search" because you should make this game's name different to that of other purchasers'. You need to create a rewarded video and an interstitial at Google Admob console. For this go to https://apps.admob.com/v2/home. Choose Apps->ADD APP and follow the instructions. Copy and paste all the generated ids into necessary xml tags. Make sure that the value of "TESTING_ADMOB_ADS" is true and enter your real Android device hash id into the "HARDWARE_DEVICE_HASH_ID_FOR_TESTING_ADMOB" field (https://developers.google.com/admob/android/test-ads). Otherwise, Google may ban your app and google play account. But don't forget to make it false before publishing your app to Play Store. I recommend you to upload an APK to internal test track Ads, IAP and other things initially (not production until you finish all the configuration).

Goggle has also changed setting up **GDPR**. If you are not familiar with the new method, go to Admob home, click the Blocking Controls on the left menu and you will see the "EU user consent" option on the new page. Click this option and configure GDPR and create a consent form there. This form will ask European users to consent or not to consent for personalized ads. If you don't setup GDPR, Google may terminate your contract. For more details, please visit: https://support.google.com/admob/answer/9770970

After these steps, your ads should work. Run the app and complete levels to see interstitial ads and tap the rewarded video button on the right-most side of the screen to view a rewarded video ad. If you don't see any ads, the most common reason is misconfiguration of GDPR. Let's try to debug the app to find the reason.

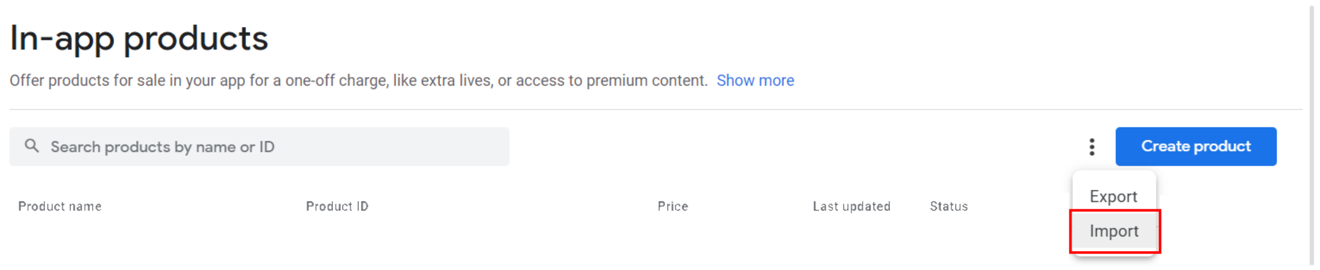
In Android Studio go to View > Tool Windows > Logcat. While your device is connected to your PC, make sure your device is selected (1), select your applicationId *if* it is there (2), Debug is selected (3), gdpr is typed at 4 in lower-case letters, Regex (5) is selected, Show only selected application is selected at number 6:



Failed to start monitoring 18762b031fcfd190 (today 3:54 PM)

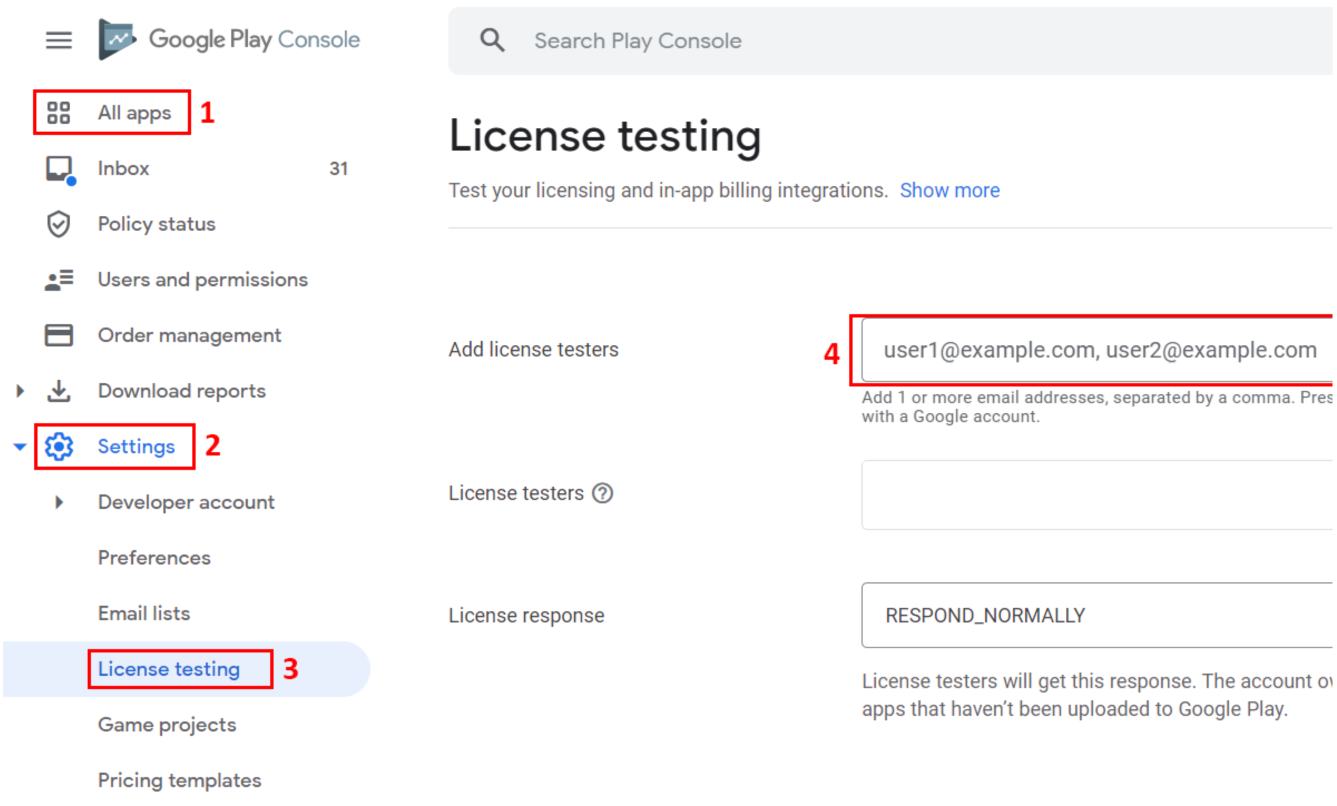
If your game is running close it and re-open it. Examine the logcat output. If you see some text that contains “onConsentInfoUpdateFailure” or “onConsentFormLoadFailure” then this means you didn't configure or misconfigured the GDPR consent settings at blocking controls. If the GDPR seems to work fine, then type “interstitial” in lowercase letters. You may see some output or you may not. If an ad fails to load the system tries to load it again after some time (you can configure this in strings.xml). So, after a while you should see some json text giving some information about the status of the interstitital ads. You can also try “rewarded” at the text input (number 4) to see the status of rewarded ads. The json output will show an error code and some explanation. Try to fix the problem according to the error message.

If you enable IAP in your game, then you should open and read the explanation in android/src/word/search/IAPActivity.java. After doing any changes in that file you should enter your IAP items into the Google Play Developer Console. But google requires your to upload an APK. Upload your APK to the *test track*. After your app is published at Play Store, in the app dashboard select your app and go to Monetize > In-app products. In zip package you downloaded from codecanyon there is an iap file that you can import to the console instead of typing each item manually (in_app_products_net.codecanyon.trimax.android.wordsearchinnovation.csv). Import this file clicking the button in red rectange as show below:

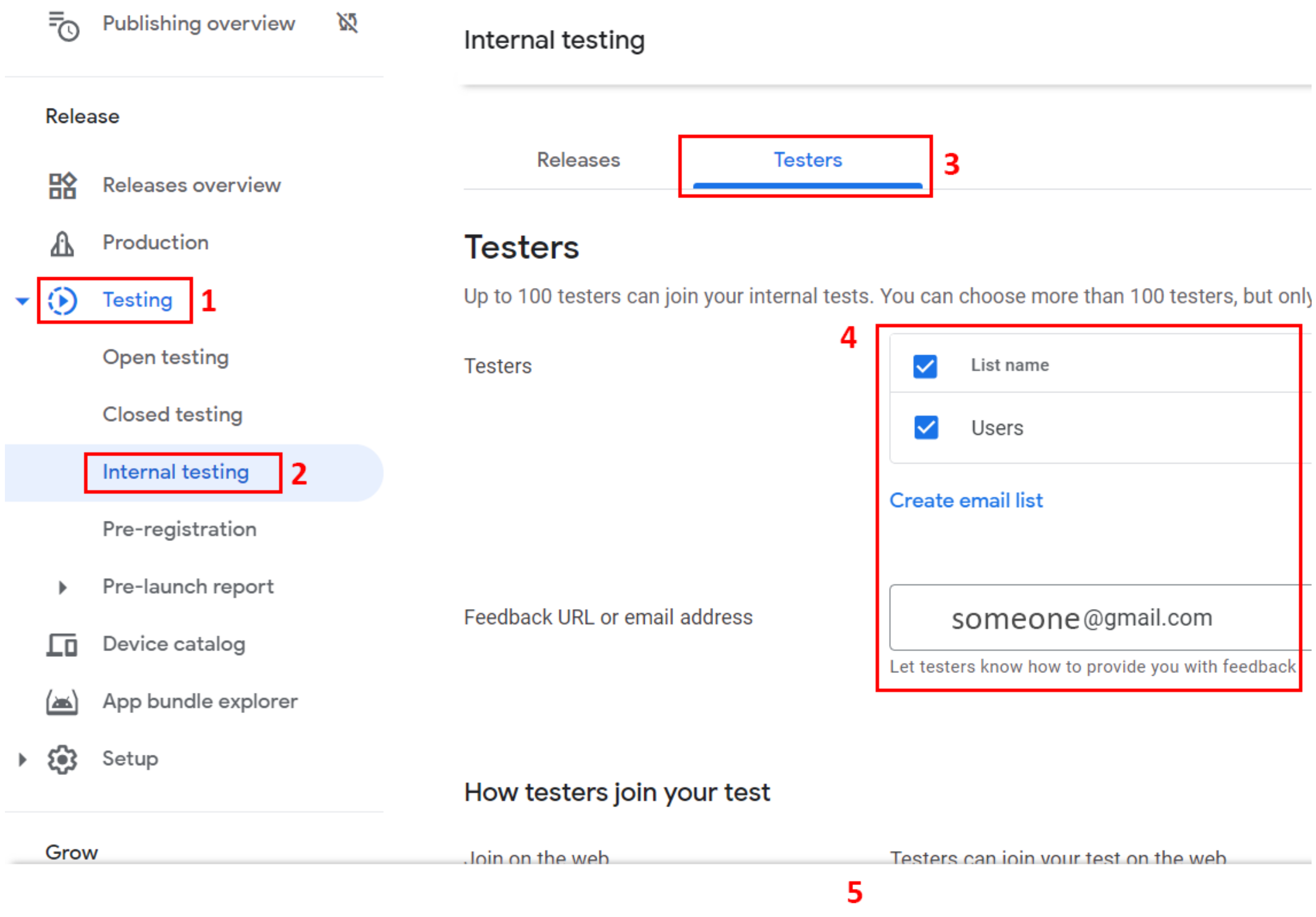

In-app products
Offer products for sale in your app for a one-off charge, like extra lives, or access to premium content. Show more
Search products by name or ID
Create product
Product name
Product ID
Price
Last updated
Status
Export
Import

However, if you added a new item or removed an existing item at IAPActivity.java, you must make your resulting in-app products list in the console **consistent** with that file. You should also be aware of the fact that you can't change or reuse an id once created. Note that you may encounter crashes in the app if you don't make the console and IAPActivity consistent.

When you are finished with it you should test it and purchase items for free without any charge. In the new Google Play Developer Console, go to All apps (1) where all your app titles are seen not specifically Modern Word Search. On the left menu choose Settings (2) > License testing (3).



On this page add a Google account email address who will do the testing (4) and save changes. Note that it may take a few hours to take effect. We are not finished yet. Now go to All apps and select your game. On the new page click Testing > Internal testing. Add the same person's account here as well and copy the app link towards the bottom of the page:



Testers

Up to 100 testers can join your internal tests. You can choose more than 100 testers, but only

Testers

✓	List name
✓	Users

Create email list

Feedback URL or email address

someone@gmail.com

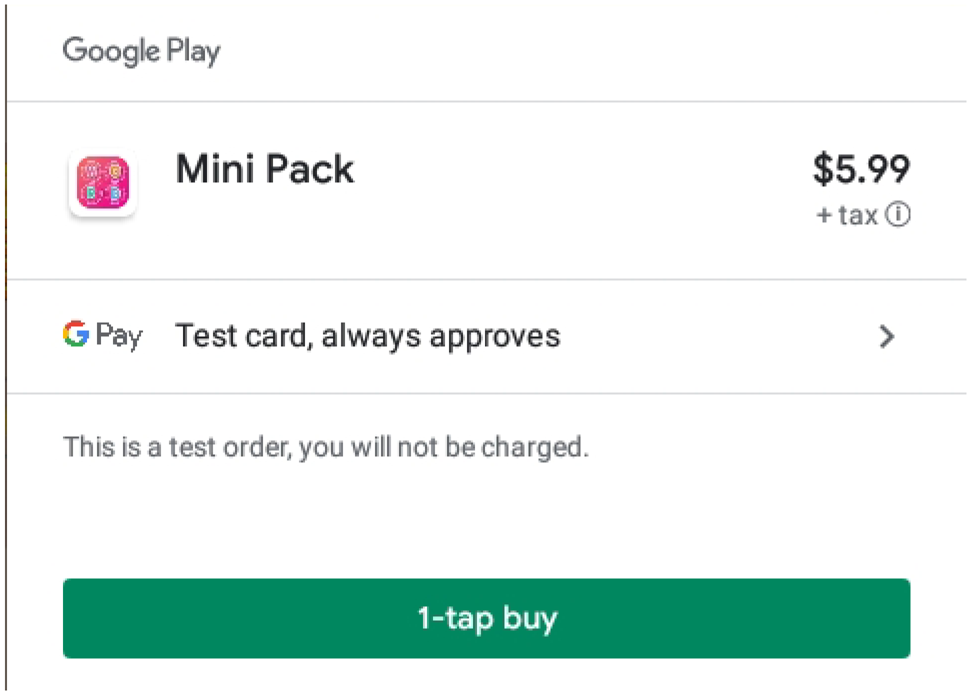
Let testers know how to provide you with feedback

How testers join your test

Join on the web — Testers can join your test on the web

The link contains the path to your private-internal word search game if you uploaded your app to the internal test track. Send this link to the test user via email. He/she must open it and agree with the testing terms and conditions of Google Play. Otherwise, he won't be able to find the app in Google Play Store App.

Assuming that you are the one doing the testing, open the shopping dialog by tapping the plus button in the game. When a user opens the shopping dialog the items titles are appended with the application's name. Google sends such a title even if you don't want it this way. To display the exact title you want, please edit the strings.properties files in android/assets/data/en or tr folders. Now, it is time to make a fake purchase. Beware that if you purchase the remove ads, you won't be able to see it again because it is a one time purchase and will be missing forever. Now, to verify that the purchase you make is free of charge, the purchase card that appears just before the transaction must be similar to the one below, i.e. it must say “This is a test order, you will not be charged”.


Google Play

Mini Pack $5.99 + tax

G Pay Test card, always approves

This is a test order, you will not be charged.

If yours is different, then review the previous steps and make sure 6-7 hours have elapsed after you uploaded your app and assigned someone to test it.

2. core/src/word/game/config/GameConfig. This is another settings file. It encapsulates the settings related to the game itself. It is easy to change the settings. If you don't have any programming experience don't worry, they are simple. true/false mean enabled/disabled. Some numerical values may have decimal places, in such cases just don't forget to append the number with f. Yes, the letter f in alphabet. That's all there is to it.

Note that after making changes in settings please test it in the game to see if it works fine and is convenient.

In the GameConfig file, There are some configuration items for firebase events. If you want to add custom events or properties for analytics, you can do so. You can call the following functions from anywhere of the game. There are static, so you don't need to an instance of an object:

```
WordGame.analytics.logEvent(String eventName, String propertyName, String propertyValue);

WordGame.analytics.logEvent(String eventName, String propertyName1, String propertyValue1,
String propertyName2, String propertyValue2);

WordGame.analytics.logEvent(String eventName, String propertyName, int propertyValue);

WordGame.analytics.setUserProperty(String name, String value);

WordGame.analytics.resetAnalyticsData();

WordGame.analytics.setAnalyticsCollectionEnabled(boolean enabled);
```

You can see these functions in android/src/word/search/FirebaseAnalyticsAndroid.java file.

If you are an experienced Java developer, you can add other Firebase analytics functions too in this file.

3. core/src/word/game/config/UIConfig. You can find various UI-related settings in this file.

4. core/src/word/game/config/ColorConfig. This file controls the look and feel of the game. There are numerous color options to change. Before changing any values don't forget the information at the top of the page.

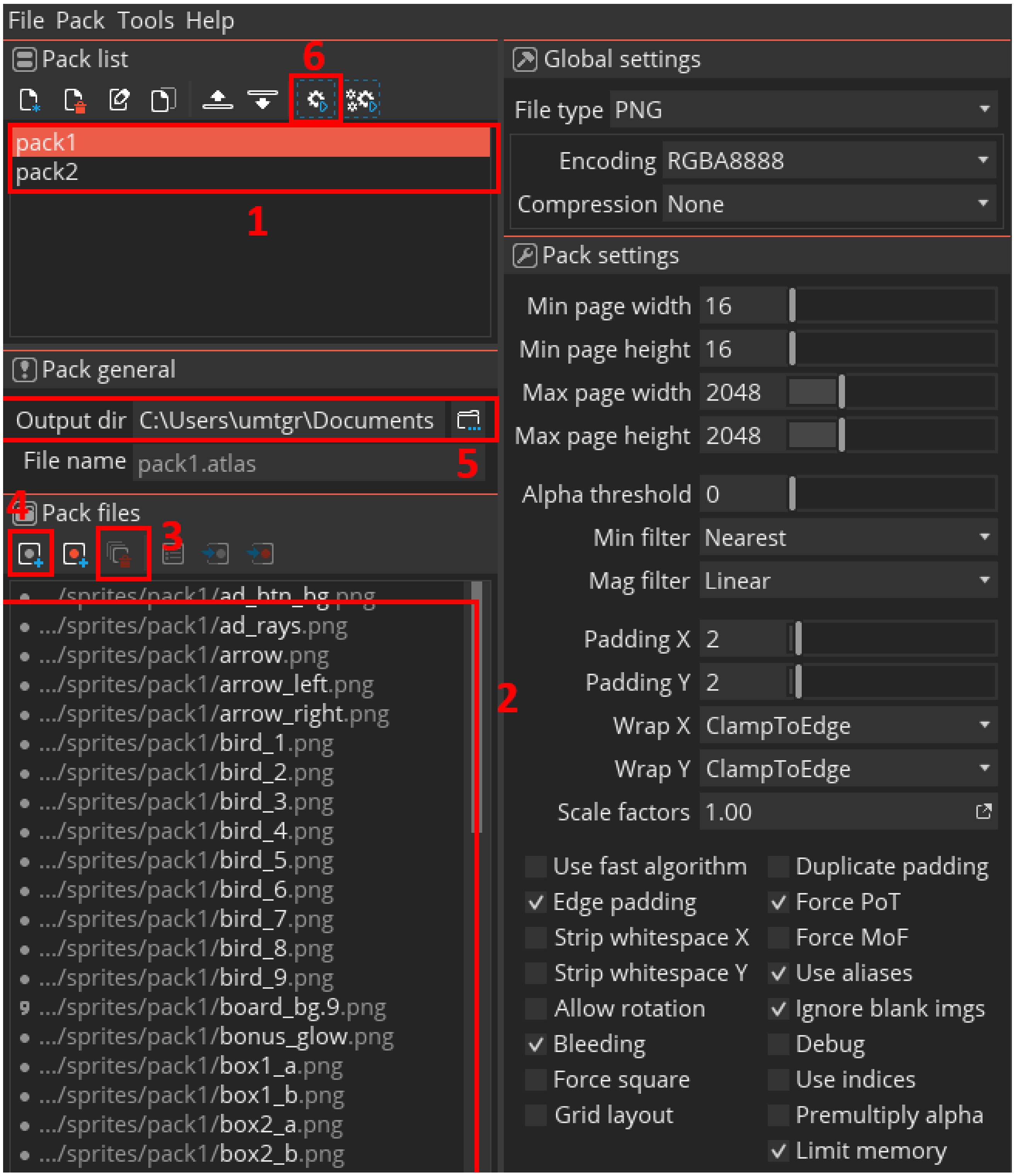
Skinning and Changing Graphics

You must change at least the game logo, icons and background images of the game. A google search will lead you to many website that are free images.

1. Game Graphics: Please note that the tools required for editing sprite sheets and fonts require Java installed on your computer. The graphics of the game are in the sprites folder in the zip file. Each folder named pack1, pack2 correspond to a sprite sheet, i.e. the contents of each folder are fused into a single image for

efficiency. The tool that packs these images is free of charge. You can download it from https://github.com/crashinvaders/gdx-texture-packer-gui/releases. Version 4.8.2 was used for this game.

Double click the “texture_pack.tpproj” file to open the sprite sheets in this program. Since the location of source image files and destination folder in your pc are different from my computer, you need to update the paths. Below, number 1 (see the next screenshot) specifies all the sprite sheets. Click the first one, select all files in the bottom pane (number 2) and delete them all by clicking number 3. Now, using the number 4, select all the images that belong to sprite sheet pack1 from your computer (from pack1 folder). Then choose the destination directory (number 5) which is YourGame/android/assets/textures. Finally, export the sprite sheet by clicking number 6:



pack1
pack2

File type PNG
Encoding RGBA8888
Compression None

Min page width 16
Min page height 16
Max page width 2048
Max page height 2048
Alpha threshold 0
Min filter Nearest
Mag filter Linear
Padding X 2
Padding Y 2
Wrap X ClampToEdge
Wrap Y ClampToEdge
Scale factors 1.00

Use fast algorithm	Duplicate padding
✓ Edge padding	✓ Force PoT
Strip whitespace X	Force MoF
Strip whitespace Y	✓ Use aliases
Allow rotation	✓ Ignore blank imgs
✓ Bleeding	Debug
Force square	Use indices
Grid layout	Premultiply alpha
	✓ Limit memory

Output dir C:\Users\umtgr\Documents
File name pack1.atlas

- .../sprites/pack1/ad_btn_bg.png
- .../sprites/pack1/ad_rays.png
- .../sprites/pack1/arrow.png
- .../sprites/pack1/arrow_left.png
- .../sprites/pack1/arrow_right.png
- .../sprites/pack1/bird_1.png
- .../sprites/pack1/bird_2.png
- .../sprites/pack1/bird_3.png
- .../sprites/pack1/bird_4.png
- .../sprites/pack1/bird_5.png
- .../sprites/pack1/bird_6.png
- .../sprites/pack1/bird_7.png
- .../sprites/pack1/bird_8.png
- .../sprites/pack1/bird_9.png
- 9 .../sprites/pack1/board_bg.9.png
- .../sprites/pack1/bonus_glow.png
- .../sprites/pack1/box1_a.png
- .../sprites/pack1/box1_b.png
- .../sprites/pack1/box2_a.png
- .../sprites/pack1/box2_b.png

Repeat this process for pack2 too. Now check the last modified date of the sprite sheet files at YourGame/android/assets/textures and verify that you did it correctly.

You should replace the images in the pack1 and pack2 folders. Image width and height values should be about the same as the original ones. Some of the images are nine-patch that are not distorted when scaled. The file names of such images end with ".9.png". If you want to replace any of such images, you should visit this address: https://romannurik.github.io/AndroidAssetStudio/nine-patches.html#&sourceDensity=640&name=example. Upload your image to this website, specify the parts of 9-patch, export the zip file and drag the highest resolution image in the zip file to the pack1 folder and overwrite the old one. If you want to add a new image to the game, add it to the pack2 folder, the spritesheet associated with that folder has a lot of empty room.

Don't change any of the settings in the texture packer program. To save memory of your users' smartphones, you should shrink the size of your sprite sheets, background and fonts. You can do this at: https://tinypng.com/

2. Icons: When you supply your icon graphics to Android studio, it creates various icons out of them automatically. But you should create your icon PNG files in photoshop first. You will create 2 files with a width and height of 512 pixels. One of the graphics must be background and the other foreground. Even if your icon is a very simple one, it must be 2 pieces as background and foreground, not 1. After you create these 2 files, in Android Studio project explorer, select Android View. Then, right-click the res folder and select New > Image Asset. Important settings should be like this for both Foreground Layer and Background Layer tabs:

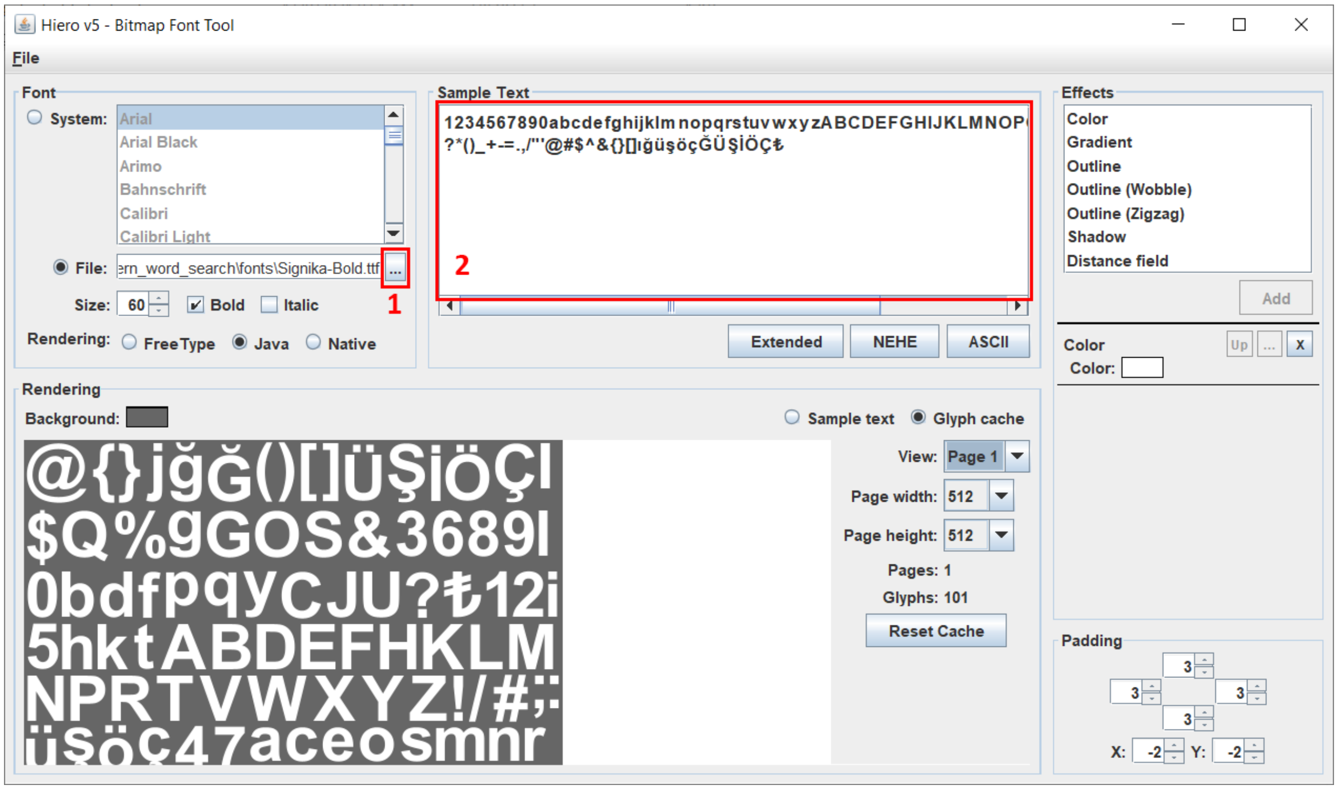
Icon Type: Launcher Icons (Adaptive and Legacy)
Name: ic_launcher
Asset Type: Image
Path: The file path to the icon image files you just created.
Show safe zone: selected (on the top-right of the window)

Now, adjust the size of your images using the Resize slider. Make sure your images are within the boundaries shown in the window. When you're done, click Next, then Finish and overwrite existing files. That's it. For more info about icons, visit: https://developer.android.com/studio/write/image-asset-studio

3. Background Images: Background images are in android/assets/bg. There are various websites that provide landscape images for free. The size of the images should be about 960x1280 so that it fits both phones and tablets. After finding new JPG files, you should reduce their file size at: https://tinyjpg.com/. You can find more info about background images at UIConfig.java settings file.

Editing Fonts

The fonts in the game are not TTF because it renders too slow. Instead, bitmap fonts are used. Libgdx has a free font generator called Hiero. Download it from: https://github.com/libgdx/libgdx/wiki/Hiero. The font related files are in the font folder (not the fonts folder in android/assets, the other one in the zip file). This folder contains the font projects and TTF files. If you want to add some new characters to an existing font, open the Hiero generator tool. There are a number of font sizes and some have shadows on them. As an example, in Hiero open “signika_bold.hiero” from the File menu. First, set the correct path to the Signika-Bold.ttf file by clicking number 1 (see the next screenshot). Then, go to the Sample Text section (number 2) and type or paste the missing characters. Afterwards, save the updated hiero file by going to File > Save Hiero settings file (it works like save-as rather than save). Finally, export the font files with File > Save BMFont files (to android/assets/fonts folder). Repeat this process for the other hiero font project files. When you are finished with one hiero project file, you should close the generator and reopen it, otherwise, the generator tends to save the current file with name of the previous project. Armed with this knowledge, you can also create new font files or replace the existing ones completely.


Hiero v5 - Bitmap Font Tool

File

Font

System: Arial

File: ern_word_search\fonts\Signika-Bold.ttf

Size: 60 [✓ Bold] Italic

Rendering: FreeType [✓ Java] Native

Sample Text

1234567890abcdefghijklm nopqrstuvwxyzABCDEFGHIJKLMNOP
?*()_+-=.,/"'@#$^&{}[]ığüşöçĞÜŞİÖÇ₺

Rendering

Background:

Sample text [✓ Glyph cache]

View: Page 1

Page width: 512

Page height: 512

Pages: 1

Glyphs: 101

Effects

Color

Color:

Padding

3 3 3 3

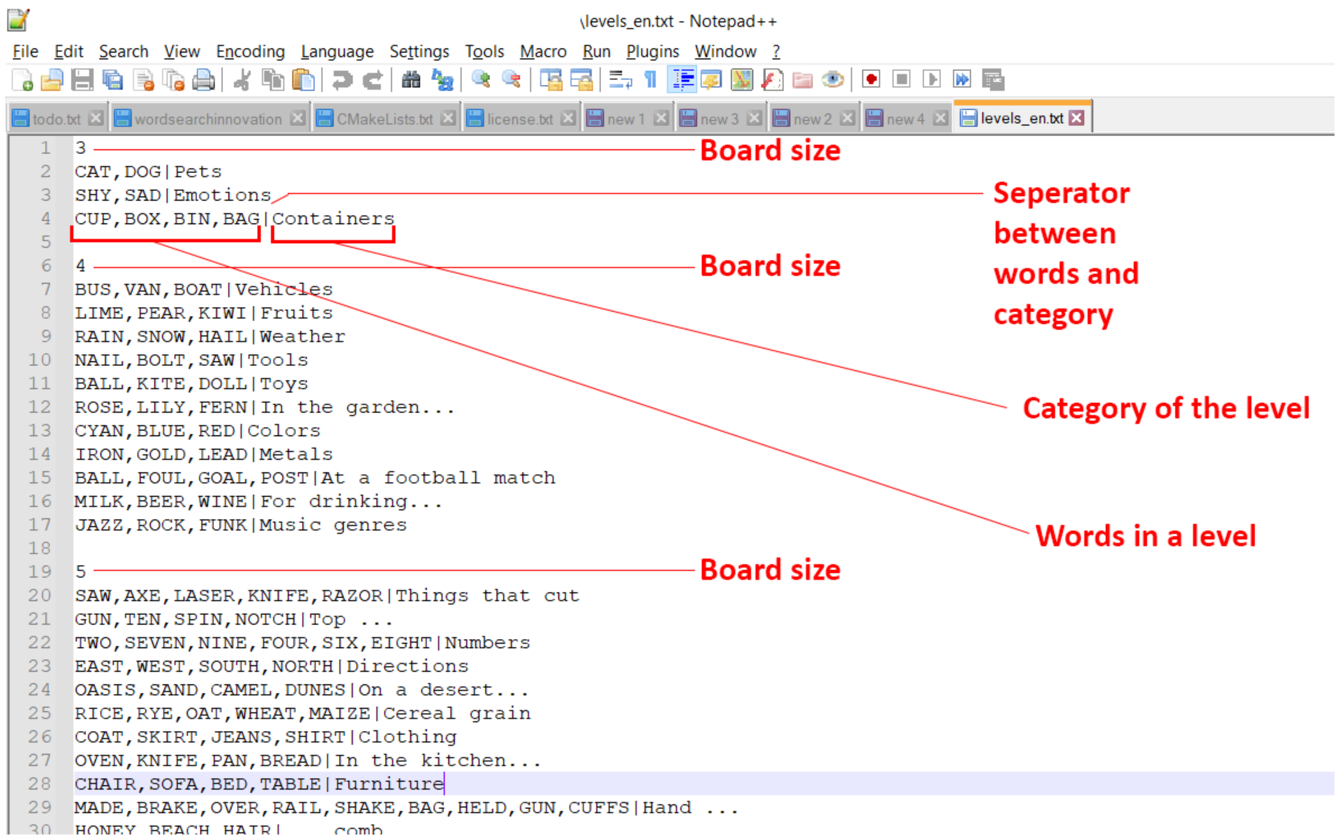
X: -2 Y: -2

Generating New Levels (And Adding a New Language)

New Levels are generated at a separate Java program that runs in Eclipse IDE. If you wish to add a new Language, keep reading.

1) First create a folder in android/assets/data and name it as the language code of your language such as “es” or “de”.

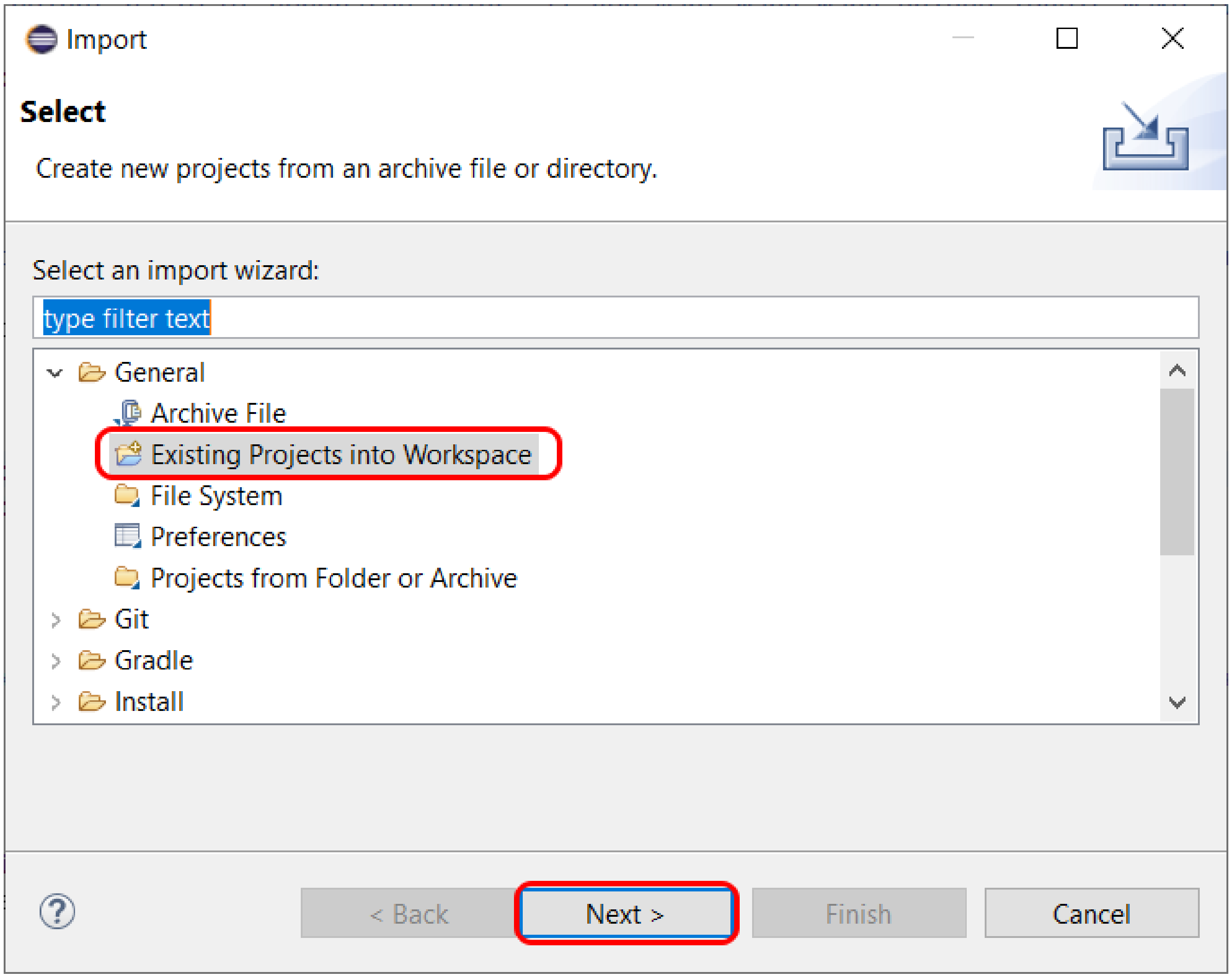
2) First you should create your levels in raw form in a temporary file. There are two raw level examples in the resources folder. One for English and one for Turkish language. Open the English raw file (levels_en.txt) in notepad++ text editor and examine it. It has a very simple structure.



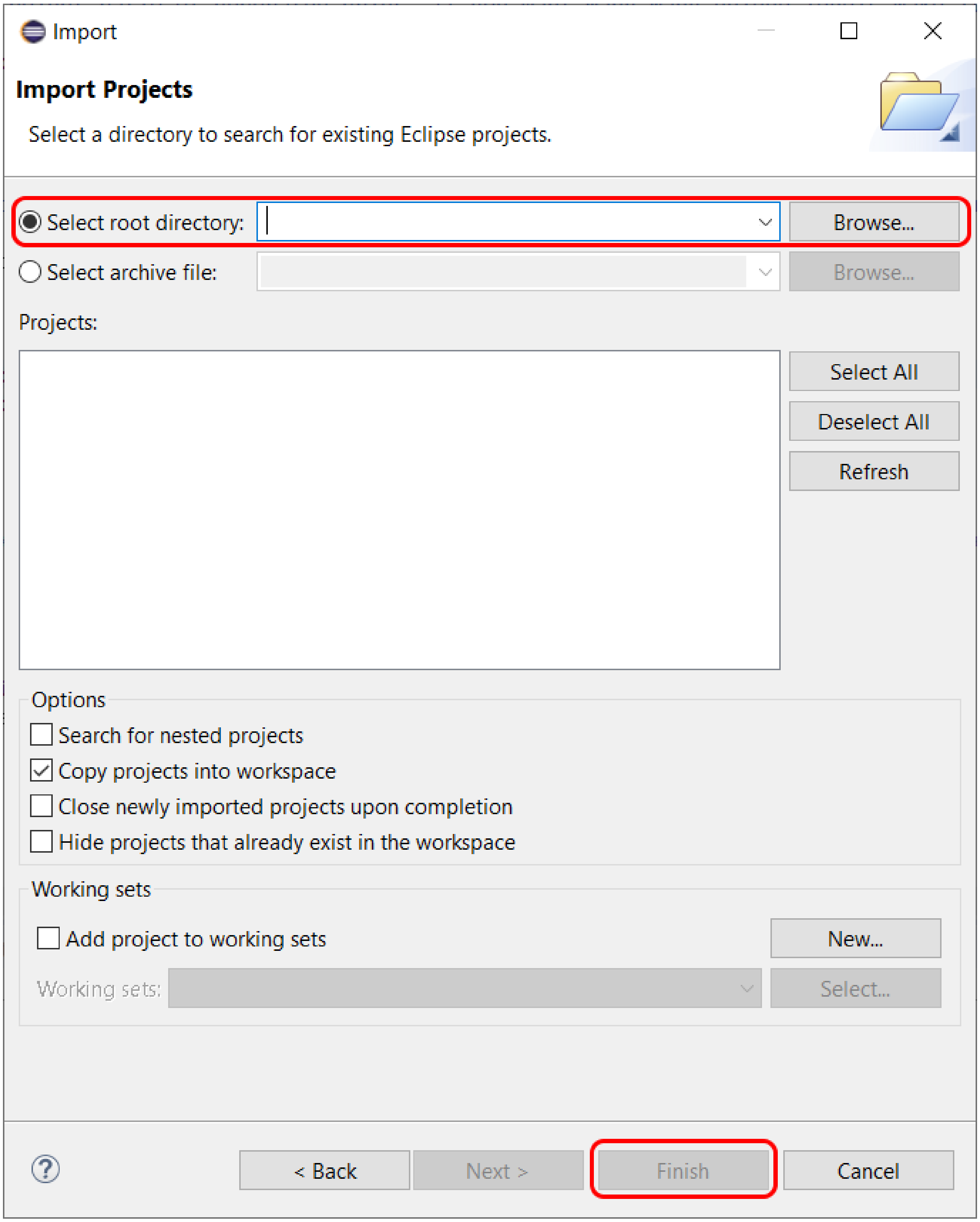
```
3
CAT,DOG|Pets
SHY,SAD|Emotions
CUP,BOX,BIN,BAG|Containers

4
BUS,VAN,BOAT|Vehicles
LIME,PEAR,KIWI|Fruits
RAIN,SNOW,HAIL|Weather
NAIL,BOLT,SAW|Tools
BALL,KITE,DOLL|Toys
ROSE,LILY,FERN|In the garden...
CYAN,BLUE,RED|Colors
IRON,GOLD,LEAD|Metals
BALL,FOUL,GOAL,POST|At a football match
MILK,BEER,WINE|For drinking...
JAZZ,ROCK,FUNK|Music genres

5
SAW,AXE,LASER,KNIFE,RAZOR|Things that cut
GUN,TEN,SPIN,NOTCH|Top ...
TWO,SEVEN,NINE,FOUR,SIX,EIGHT|Numbers
EAST,WEST,SOUTH,NORTH|Directions
OASIS,SAND,CAMEL,DUNES|On a desert...
RICE,RYE,OAT,WHEAT,MAIZE|Cereal grain
COAT,SKIRT,JEANS,SHIRT|Clothing
OVEN,KNIFE,PAN,BREAD|In the kitchen...
CHAIR,SOFA,BED,TABLE|Furniture
MADE,BRAKE,OVER,RAIL,SHAKE,BAG,HELD,GUN,CUFFS|Hand ...
HONEY,BEACH,HAIR|     comb
```

Each level goes to one line in the file. First words are written, seperated by a comma and in capital letters. After the words, a pipe character (|) is used to end the words. After the pipe character, the category of the level is written. That's it! The numbers specify the number of the board, for example, 3 is 3x3, 4 is 4x4. So numbers group different board sizes. The minimum size is 3x3, maximum 8x8. Higher than 8 was not tested. There can be empty lines (which are ignored). Make sure your levels file is UTF-8 encoded. In notepad++ this is done at Encoding > Convert to UTF-8.

3) After you create your levels, download the Eclipse IDE at https://www.eclipse.org/downloads/. Set it up or unzip it to a suitable location on your PC. Once ready, go to File > Import... This will lead you to the following:



Make sure the option “Existing Projects into Workspace” is selected and click Next. In the next dialog box select the root folder of the Eclipse generator project folder named WordSearchGenerator. This folder is in the zip files you downloaded from Codecanyon. Then click finish:


Import

Import Projects

Select a directory to search for existing Eclipse projects.

[✓ Select root directory:] Browse...

Select archive file: Browse...

Projects:

Select All

Deselect All

Refresh

Options

- [ ] Search for nested projects
- [x] Copy projects into workspace
- [ ] Close newly imported projects upon completion
- [ ] Hide projects that already exist in the workspace

Working sets

- [ ] Add project to working sets

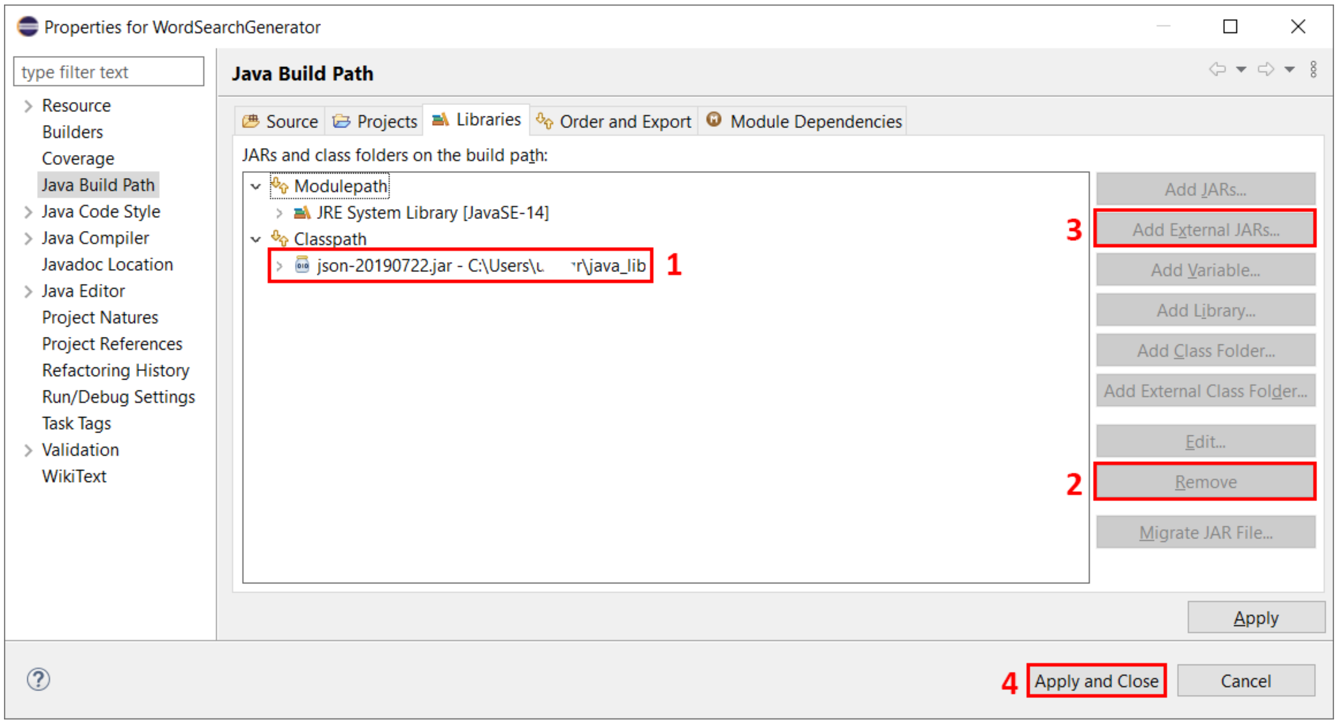
New...

Working sets: Select...

< Back | Next > | Finish | Cancel

After this process, the project will be placed on the left part of Eclipse IDE.

However, an error will pop up because it can't find a dependency it needs. In the zip file you downloaded from Codecanyon, there is a folder called eclipse_dependencies, there is a file in this folder that we should make the Eclipse project find. In the project pane, right click the top project folder and choose Build Path > Configure Build Path. Such a dialog will open:


Properties for WordSearchGenerator

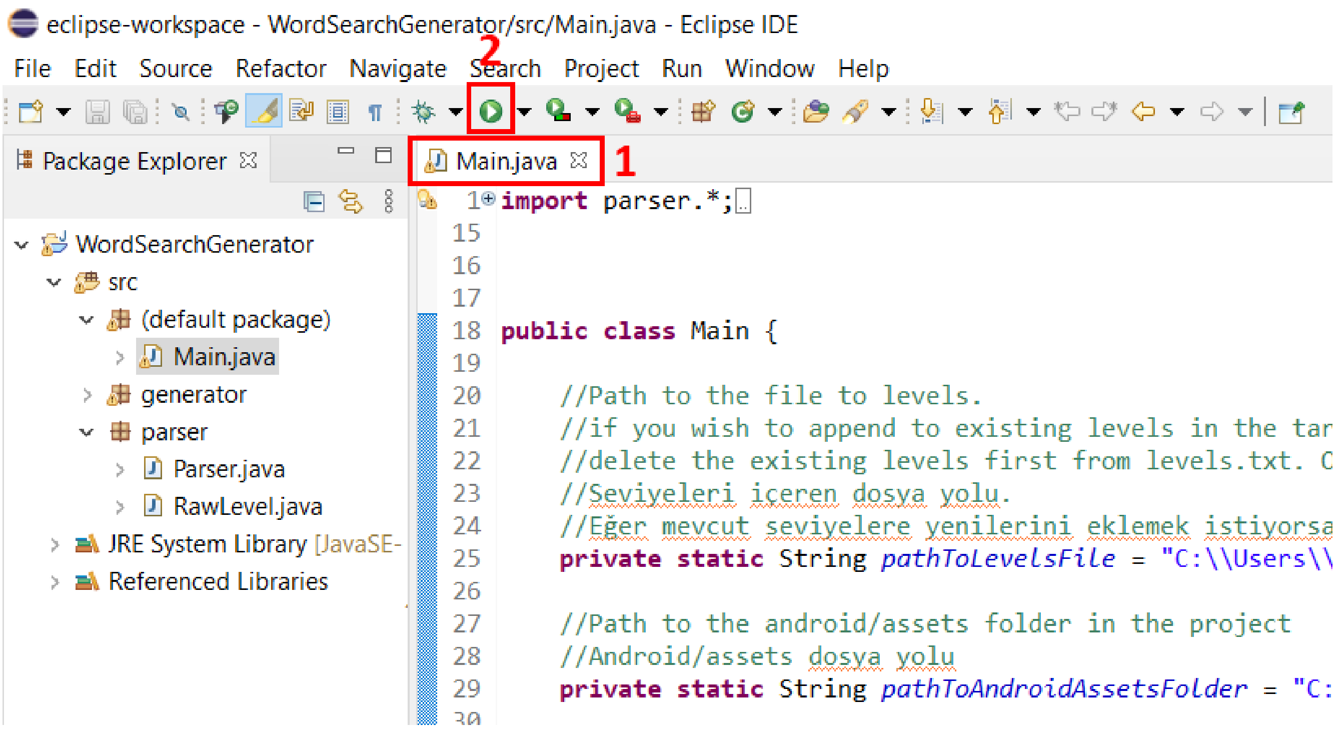
Java Build Path

JARs and class folders on the build path:

- Modulepath
  - JRE System Library [JavaSE-14]
- Classpath
  - json-20190722.jar - C:\Users\u r\java_lib

Select the problem file (1) and remove it (2). Then, select the dependency file on your hard drive clicking 3 and finally click Apply and Close (4). The error should disappear.

In the project pane, open src/default package/Main.java. There are configuration options on this page and they are fully commented. Change them according to your needs. Everything should be ready to generate the levels now. The generator tool exports each level in its own file rather than all in one, to save memory at run time. Open the Main.java (1) file mentioned before and run (2) the program:


eclipse-workspace - WordSearchGenerator/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
2
Package Explorer
Main.java
1
WordSearchGenerator
src
(default package)
Main.java
generator
parser
Parser.java
RawLevel.java
JRE System Library [JavaSE-
Referenced Libraries

```
import parser.*;


public class Main {

    //Path to the file to levels.
    //if you wish to append to existing levels in the tar
    //delete the existing levels first from levels.txt. C
    //Seviyeleri içeren dosya yolu.
    //Eğer mevcut seviyelere yenilerini eklemek istiyorsa
    private static String pathToLevelsFile = "C:\\Users\\

    //Path to the android/assets folder in the project
    //Android/assets dosya yolu
    private static String pathToAndroidAssetsFolder = "C:
```

You can see information about the generated levels in the console output. **This is important: When you finish generating levels, you must inform the game about the total number.** Go to Android Studio, open core/src/word.search /config/GameConfig and find the availableLanguages variable. If you generated these levels for an existing language, update the number of levels (levelCount) parameter with the new total level count. If these levels are for a new language, then enter the number of levels along with other language info as a new "put" statement. Be careful: don't enter the number of the final level, total number is final level index + 1.

4) When you add a new language into the game, you should also come up with a translation file for user interface. Go to android/assets/data/en, copy strings.properties and paste it into your new language folder you created in step 1. Open it in Android Studio or another text editor. If you create a brand new strings.properties file, don't forget to make its encoding **UTF-8**, otherwise your non-ascii characters may not be visible on GUI. You shouldn't delete the numerical parameters that take place in curly braces while translating the text.

5) When you add a new language, you will also create some images for this new language. Remember the section about creating sprite sheets? If you you haven't read that section (Skinning and Changing Graphics). In that section we talked about the sprite folder called pack2. Now open that folder. There 3 different types of textures in that folder:

-cup_text_en.png

-logo_en.png
-stamp_en.png

These files have text that is exlusive to a language. In the psd folder find the corresponding psd files and open them in Photoshop, edit and export them to the pack2 folder. Then import this new image into the texture packer program. Don't forget to put your new language's two-letter code at the end of file names. When you're finished don't forget to export the resulting spritesheet (to android/assets/textures).

6) The dictionary. When a user finds a word and taps on it, a dictionary dialog opens with a definition of that word. There may be very-little used, obscure words and when players don't know the meaning, they wish to learn them. Therefore, most word games offer such a functionality. Our word connect game has a dictionary for the English language only. The dictionary service is free, has some hourly or daily limit. To be able to use the dictionary, you must sign up with the Wordnik at: https://developer.wordnik.com/ Sign up and get an API key. When you acquire your API key, make it the value of WORDNIC_API_KEY at GameConfig.java file. If you don't sign up, your app may crash. If you don't want to use the service you can completely disable this feature at GameConfig.java by making ENABLE_ENGLISH_LANGUAGE_DICTIONARY false.

Adding an offline dictionary for Turkish or any other language is not an option because it may raise the file size of APK significantly.

COPYRIGHT INFORMATION

Product	**License**
Libgdx game engine	Apache 2
JSON	Json license
Worknik: https://developer.wordnik.com/terms	Attribution license
Signika font	Google open source license
Eclipse IDE	Eclipse Public License
coins: https://pngtree.com/freepng/a-bunch-of-coins_4087919.html	Free license (with attribution)
gift box: https://pngtree.com/freepng/happy-	Free license (with attribution)

birthday-ballons-with-birthday-gift-box-png_5316014.html	
daily gift boxes: https://www.freepik.com/free-vector/boxing-day-sale-background_3337284.htm#page=10&query=gift+box&position=29	Free for personal and commercial use (with attribution)
dictionary icon: https://www.flaticon.com/free-icon/dictionary_1902726?term=dictionary&page=1&position=3	Settings icon https://www.flaticon.com/free-icon/settings_148912?term=settings&page=1&position=19
light bulb: https://www.freepik.com/free-vector/bulb-icon_4790281.htm#page=1&query=bulb&position=13	Free for personal and commercial use (with attribution)
magnifier: https://pngtree.com/freepng/hand-drawn-yellow-handle-magnifier_4546693.html	Free license (with attribution)
magic wand: https://pngtree.com/freepng/pack-of-halloween-flat-icons_5666877.html	Free license (with attribution)
shop tent: https://www.freepik.com/free-vector/realistic-set-striped-awnings-isolated-background-clipart-with-red-white-tents_3586282.htm	Free for personal and commercial use (with attribution)
padlock: https://pngtree.com/freepng/golden-lock-icon-vector_5066545.html	Free license (with attribution)
red carpet: https://pngtree.com/freepng/white-winners-podium-vector-red-carpet-stage-for-awards-ceremony-illustration_5205015.html	Free license (with attribution)
Background images https://pngtree.com/freepng/blue-hand-painted-fresh-landscape_4109367.html	Free license (with attribution)

https://pngtree.com/freepng/monthly-sky-starry-sky-landscape_4119610.html https://pngtree.com/freepng/original-sunny-beach-blue-landscape-gradient-illustration_4218485.html https://pngtree.com/freebackground/scenic-nature-background_259993.html https://pngtree.com/freebackground/grass-sky-landscape-field-background_356048.html https://pngtree.com/freepng/illustration-camping-night-landscape_4108672.html	
Background images https://www.freepik.com/free-photo/image-scenic-beach-es-torimbia-toranda-asturias-spain_10399428.htm https://www.freepik.com/free-photo/fresh-grass-with-sky-background_912961.htm https://www.freepik.com/free-photo/dandelions-meadow-summer-day_1473678.htm https://www.freepik.com/free-vector/stone-well-nature-night-scene_11863150.htm https://www.freepik.com/free-vector/hilly-mountains-landscape-seasons-spring-summer-autumn_12417691.htm	Free for personal and commercial use (with attribution)