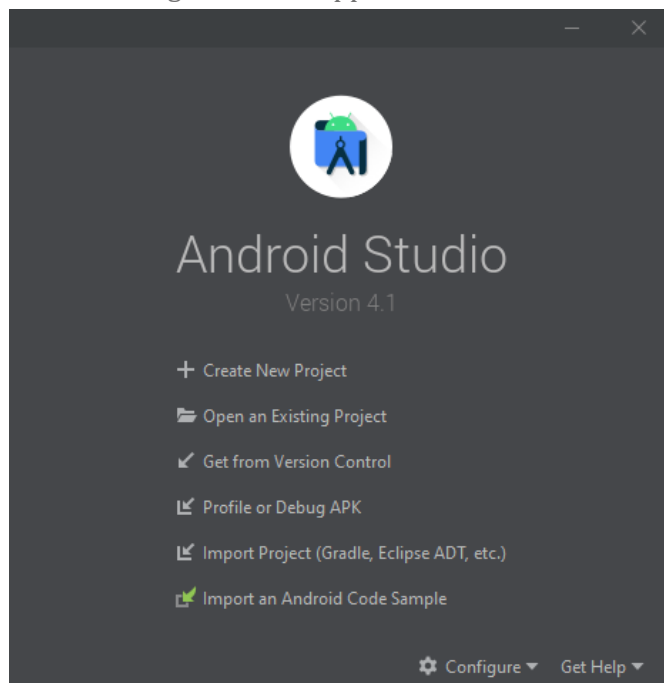


VISIDA LOCALISATION GUIDE


SETTING UP THE ANDROID STUDIO PROJECT

1. Navigate to <https://developer.android.com/studio> and download Android Studio.
2. Run the installer. An install wizard will appear. Click next until the end of the install process.
3. Next, launch Android Studio. A second setup wizard will appear. Again, progress through this wizard using the default settings.
4. The following menu will appear:



Select “Get from Version Control”.

5. Enter the following URL and select “clone”:
To get the link from Bitbucket, open the main page of the project on Bitbucket and copy the link from here:



Take the next steps for this new repository and its freshly added files

Copy and connect the repository locally so that you can push updates you make and pull changes others make. Enter **git clone** and the repository URL at your command line:

```
git clone https://JohnHoffman2@bitbucket.org/ConnorDodd/visida.git
```

[Learn more](#) or [clone in Sourcetree](#) to avoid the command line. [Sourcetree](#) is a free Git and Mercurial client.

✕

This step will be different when we release this guide as it will then be a

public link. (you will also be prompted with your Bitbucket password here).

6. After this is done it will take the project a small amount of time to initialise. Wait until the loading in the bottom toolbar is complete.
7. Now we have to switch to the configurable version of the app. To do this, select “VCS” in the top toolbar of Android Studio, the “Git”, then “branches”.
8. In the “remote” section, select “origin/release/configurable”, then “checkout”.

TEXT TRANSLATIONS

1. In the Android studio project files (located in the project tab of Android Studio), navigate to res/values/strings. This folder holds the translated text for each localization.
2. When opening the “string.xml” file with no brackets after it, the following banner should appear across the top:

Edit translations for all locales in the translations editor. [Open editor](#) [Hide notification](#)

3. The “open editor” button opens a graphical interface where translations for each string of text can be updated. Note that not all translations can be edited using the graphical interface. Some translations are stored in arrays:

```
<string-array name="preference_values_eat_icon">
  <item>ic_btn_eat_knife_fork</item>
  <item>ic_btn_eat_spoon</item>
  <item>ic_btn_eat_chopstick</item>
  <item>ic_btn_eat_hand</item>
  <item>ic_btn_eat_table</item>
</string-array>
```

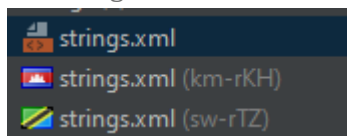
These translations must be edited in the respective xml files shown in step 5.

4. To add a localization that does not yet exist in the app, select the following icon (in the graphical editor):



This will add a column where translations can be added and edited.

5. You will notice that this has created a new file in res/values/strings. Make note of the tag after the file which is Android’s identifier of the localization:



This tag will be needed in the following stages.

6. We must also add this new language to the configuration menu so that it can be set in the app. The following two arrays in the “strings.xml” must be added to:

```

<string-array name="preference_language_list">
    <item>English</item>
    <item>Khmer</item>
    <item>Swahili</item>
</string-array>
<string-array name="preference_language_entryvalues">
    <item>en</item>
    <item>km</item>
    <item>sw</item>
</string-array>

```

The label in the first array can be anything. This is how it will appear in the app:

App Language

- ☐ English
- ☐ Khmer
- ☐ Swahili

The second label must be the first two letters of the tag identified in step 5. The app uses this to set the language.

INSTRUCTION TRANSLATIONS

1. Text, audio, and an image must be provided for each individual instruction.
2. To add the images to the project, first locate the res/drawable folder, right click on it, and select "Show in Explorer".
3. Create a new folder here and name it "drawable-<localization>-nodpi" where <localization> is the tag from step 6 in the text translations. An example folder name is "drawable-sw-rTZ-nodpi".
4. Images, in png format, should be placed in the folder. Try to give the images descriptive names that indicate they are an instruction and the instruction they provide. An example image name is "inst_delete_dialog.png".
5. To add the audio files to the project, first locate the res/raw folder, right click on it, and select "Show in Explorer".
6. Create a new folder here and name it "raw-<localization>" where <localization> is the tag from step 6 in the text translations. An example folder name is "raw-sw-rTZ".
7. Audio, in mp3 format, should be placed in the folder. Try to give the files descriptive names that indicate they are an instruction and the instruction they provide. An example image name is "inst_delete_dialog.mp3".
8. To add a new localization with instructions, locate the res/values/instructions_text folder.
9. Right click on this folder and select "New", "Values Resource Directory".
10. In the prompt that appears, name this file "instructions_text" and select "Locale" in the list of available qualifiers. This will provide the options to set the language and region for the new instruction material.
11. Instructions can be added to the file in the following format:

```
<string-array name="instruction_activityname">
    <item>
        Your instruction text here.
    </item>
    <item>
        As many additional instructions that are needed can be
        added.
    </item>
</string-array>
```

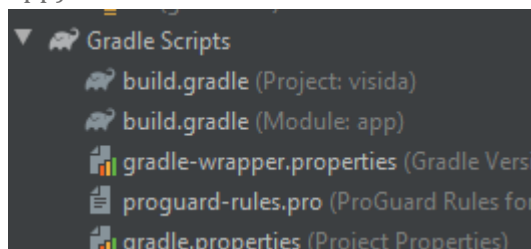
12. The name in the string array refers to the pages of the app that this instruction will be displayed. A list of these names and the screens that they correspond to can be found in **Appendix – need to add this**.
13. To add the media files, locate the instructions_media folder and repeat steps 2 and 3, to create a file named “instruction_media” for the same locale.
14. Media is added to this file in the following format:

```
<array name="inst_media_activity">
    <item>@drawable/inst_image_name_1</item>
    <item>@drawable/inst_image_name_2</item>
</array>
<array name="inst_media_activity_audio">
    <item>@raw/inst_audio_name_1</item>
    <item>@raw/inst_audio_name_2</item>
</array>
```

15. The image and audio paths need to correspond to the files we added in steps 4 and 7. The text, images and audio in each array need to align. This means that the first text item, the first image and the first audio in an array will be part of the one instruction. Note that the number of text items, images and audios for each screen must be the same, otherwise the app will crash when loading the instructions.

BUILDING THE APK

1. (Optional) The size of the built apk can be reduced by only including one set of localization files. This does mean that the language of the app cannot be changed to anything but this localization and English. Locate the build.gradle (Module: app) file.



In the “defaultConfig” section add the following tag:

```
resConfigs "<localization>"
```

Where <localization> is the tag from step 6 of the text translations.

2. Select “Build” from the top menu bar and select “Generate Signed bundle / APK”. This will open a wizard to build the apk.
3. The key store path should be set to the root folder of the project as follows “.....\visida\VISIDA_Keystore.jks”. The start of the file path will depend on where you have the project saved.
4. Enter “visida” as the “Key store password and the “Key password”.
5. Select “release”, then “finish”.
6. The built apk will be located in “ \visida\app\release”.