# What To Do: Developer's Manual

This document contains an overview of the What To Do app, necessary specifications and a guide how to get started.

## Getting started

Clone the repository manually or using:
git clone https://github.com/emeliehedqvist/whattodo

## Android SDK targets

Minimum SDK: 19
Target SDK: 22

## Google Maps Api Key

To be able to use the maps function of the application, you need a Google Maps Api Key. To get one you need to find your signing certificate fingerprint (SHA1).The easiest way to obtain a Google Maps Key is to follow these few steps.

Using Android Studio:

1. Create a new project with a Google Maps Activity
2. Open google_maps_api.xml, where the SHA1 is specified
3. Follow the instructions but change the package name to com.emehed.emeliehedqvist.whattodo
4. Paste the key obtained as the value of the string in google_maps_api.xml

## Google Maps

If the key was obtained and added correctly the map view will display your current location as well as a marker of the suggested place. Otherwise it will display a blank page with a Google logo at the bottom.

## Google Places

The What To Do app uses Google Places to search for and display places which are later recommended to the user. The app performs a Nearby Search Request on Google Places API Web Services. More information of how to perform and use the search request can be found on the following page.

https://developers.google.com/places/webservice/search

## Testing

Throughout the development phase the app has continuously been tested by the developers. The testing has been done manually and performed on both simulators and different physical devices to see if the app behaved as expected. When errors or bugs were discovered the logs in Android Studio were consulted to see where the errors originated from and what the problem was. The manual testing helped us to see the app from the user's perspective and also gave us more ideas for improvement apart from helping us detect bugs.

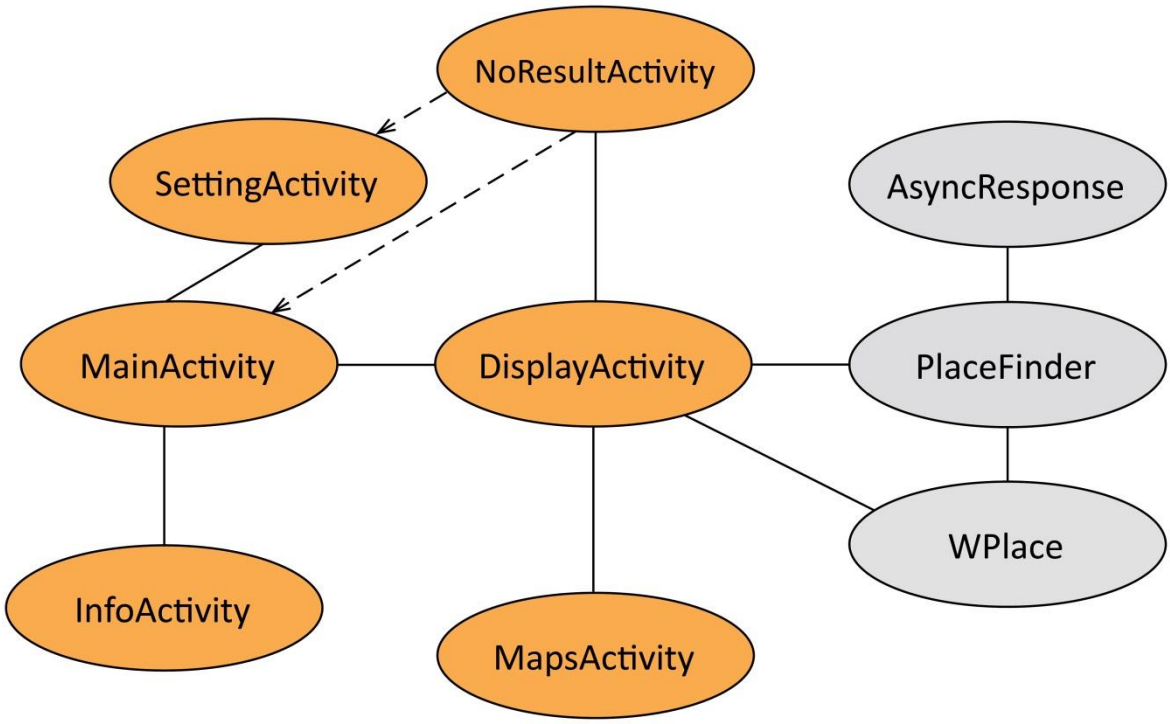
Unfortunately, no automatic testing was performed. If we were given the opportunity to

redo this project we would have tried to use some form of automatic testing. We would have liked to try out JUnit or Robotium to further test our application and complement the manual testing.

The application has been tested on the following physical Android devices:
Samsung Galaxy S4 mini - Android 4.4.2
Samsung Galaxy S5 - Android 5.0
Google Nexus 7 - Android 5.1

## Architecture Specification

The app contains six activities, two other classes and one interface. The image below shows how the relations between the classes. For further description of what the classes and their methods do the reader is directed to the source code which includes more detailed comments




## Future

There are areas of the app that warrant further development. The first step in the next development phase should be to implement SensorListener in DisplayActivity as a way to make it possible to do a new search by shaking the application. At the same time it would be convenient to add a button as an alternative to perform a new search if the recommended place is unwanted. There is also room for refinement in the selection process of the randomizing algorithm that selects a place from an array in PlaceFinder. Instead of only selecting a place and returning it an improvement could be to remove the place from the list and retaining the list to allow faster search if the recommended place is rejected by the user. One other functionality that has been discussed during the

development phase is to add easy ways to share content to social medias such as Facebook or Twitter.