# Android App Development

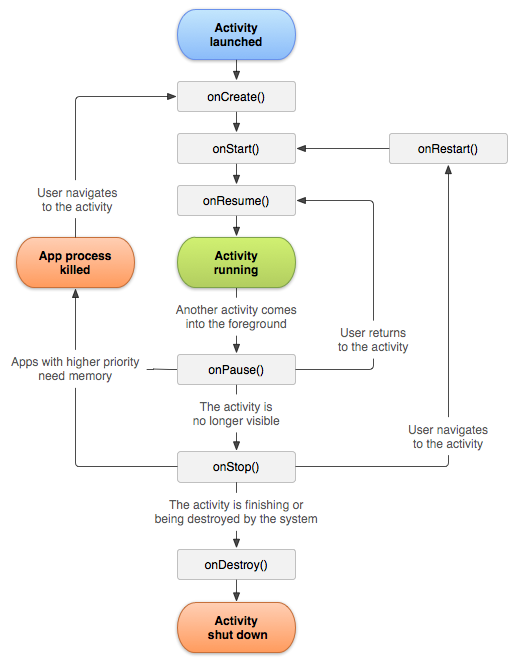
The Bananco application is implemented for Android as Android App. The Smartphone Sensing Framework was used, which is already capable of monitoring various sensors, such as acceleration, orientation and magnetic fields. The framework is also capable of displaying a continuous video stream from the camera and taking samples to be processed by a classifier network. To be able to build the Bananaco application using the Smartphone Sensing Framework, one needs to understand the basics in Android App development.

## Manifest

The AndroidManifest.xml contains for the operating system relevant information. For example, required permissions, as well as required features are listed here. In addition, all invokable services and activities are also listed.

In the case of the Bananoco application, the permission to access the camera is listed as ‘uses-permission’ and the main activity is listed as entry-point for the application.

## Activity-Lifecycle

In Android the application must provide an implementation that extends the Activity class and the application must adhere to the Activity Lifecycle (see Figure XX).

As the graph shows, there are multiple steps before and after the actual running state. The states before (onCreate, onStart and onResume) are used to prepare or resume resources required in the running state. For example, at the onCreate stage and in the ImageDetection activity, the camera is activated and the app prepared to be able to process incoming images. The stages onPause, onStop and onDestroy might be invoked after the running stage to tell the application to free resources or to stop processing data. For example, the onPause callback in the Bananaco application ensures that no more images from the camera are retrieved or classified to reduce battery usage.

While onCreate and onDestroy mark the lifetime boundaries of the beginning and end of the activity, the other stages marked by the callbacks are not as trivial. Once an app is no longer visible to the user, the method onStop is called, which allows the app to free all UI-relevant resources and to stop foreground tasks. It also allows the app to actively maintain remaining tasks as background tasks, such as continuing to download game assets. A call of onRestart followed by onStart allows the application to re-allocate UI-relevant resources and to prepare for user interactions. The method onPause tells the application that it is currently no longer in the foreground (another activity is hiding it) and not seen by the user. It is useful to pause foreground tasks to reduce until onResume is called, because those results are not visible to the user.

## Android Native Development Kit

References

Activity Lifecycle:

<https://developer.android.com/guide/components/activities/activity-lifecycle>

<https://developer.android.com/guide/components/images/activity_lifecycle.png>