System Design Document for Barcode Scanner Project (SDD) - Group 30

Christian Svensson Olle Andreasson Olof Karlsson
Rasmus Letterkrantz

May 26, 2013

Contents

1 Introduction

1.1 Design Goals

Our goal with the structure of our code is to keep the database, the algorithm for analyzing an image and the android specific GUI, separated from each other so that each and every one of these parts can be exchanged or updated without affecting the performance of the other. The way that Android applies MVC[1] is that it forces you to create an Activity for each view you want to display. Each activity requires an accompanying XML file. The XML file defines the user interface and the Activity is the controller for that view. The only way to add data to this is by adding you own (Java) code and forwarding that data to the controller. In other words, MVC is applied for each new view you create for your application.

1.2 Definitions, Acronyms and Abbreviations

- GUI, Graphical User Interface.
- Android, open source operating system for mobile units (smart-phones, tablets, etc).
- Java, platform independent programming language.
- XML, Extensible Markup Language is used for structuring the elements in the GUI.
- SQLite, the database used to store the barcodes.
- Locator, class used to locate the barcode on a given image.
- Generator, class used for analyzing the pattern extracted by the Locator and generating the actual key.
- MVC, a way of partition an application with a GUI into distinct parts avoiding a mixture of GUI-code, application code and data spread all over.

2 System Design

2.1 Overview

The application will strive to keep a MVC way of structuring the code but as the standards for programming Android have a very unique way separating views and data this will be more reflected in the functionality rather than the visual appearance of the code.

2.1.1 The Model Functionality

CameraActivity's interface will be the main entry for our model functionality. From there the different functionality parts of our model can be called and then presented in other, separated views.

2.1.2 Storage

Each barcode will be stored in a SQLite database as a product. The barcode itself will be used as the identifier of the product and the price, description and name of the product will be stored along with it. The database will be using a single table and view called Products and ProductsView, respectively.

2.2 Software Decomposition

2.2.1 General

The application is decomposed into the following parts:

- camera, contains the CameraActivity and a helper class to the camera.
- activities, contains all the other activities.
- core, contains all of our logic (the barcode scanning algorithm).
- database, contains all classes that handle the database.

2.2.2 Decomposition into subsystems

The only subsystems in the application is the DatabaseHandle which writes and reads to/from the database, and the camera which is used to take an image.

2.2.3 Layering

On our highest level we have the views that are displayed on the screen. Following those we have the activities which are controlling what the views are going to display. On our lowest layer we have the core logic (analyzer, DatabaseHelper, etc) which is read by the activities.

2.2.4 Dependency analysis

The dependencies that exists are shown in diagram below. All other dependencies not listed are the ones between Activity -> View and Activity -> Model; in other words the fact that the Activity needs the data from the model and the view to display the data on from the interface.

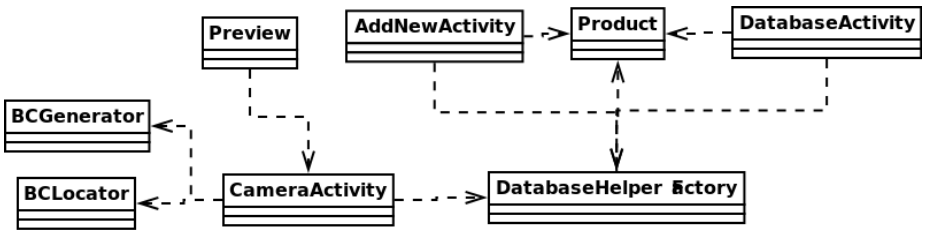

Figure 2.1: Dependency Diagram

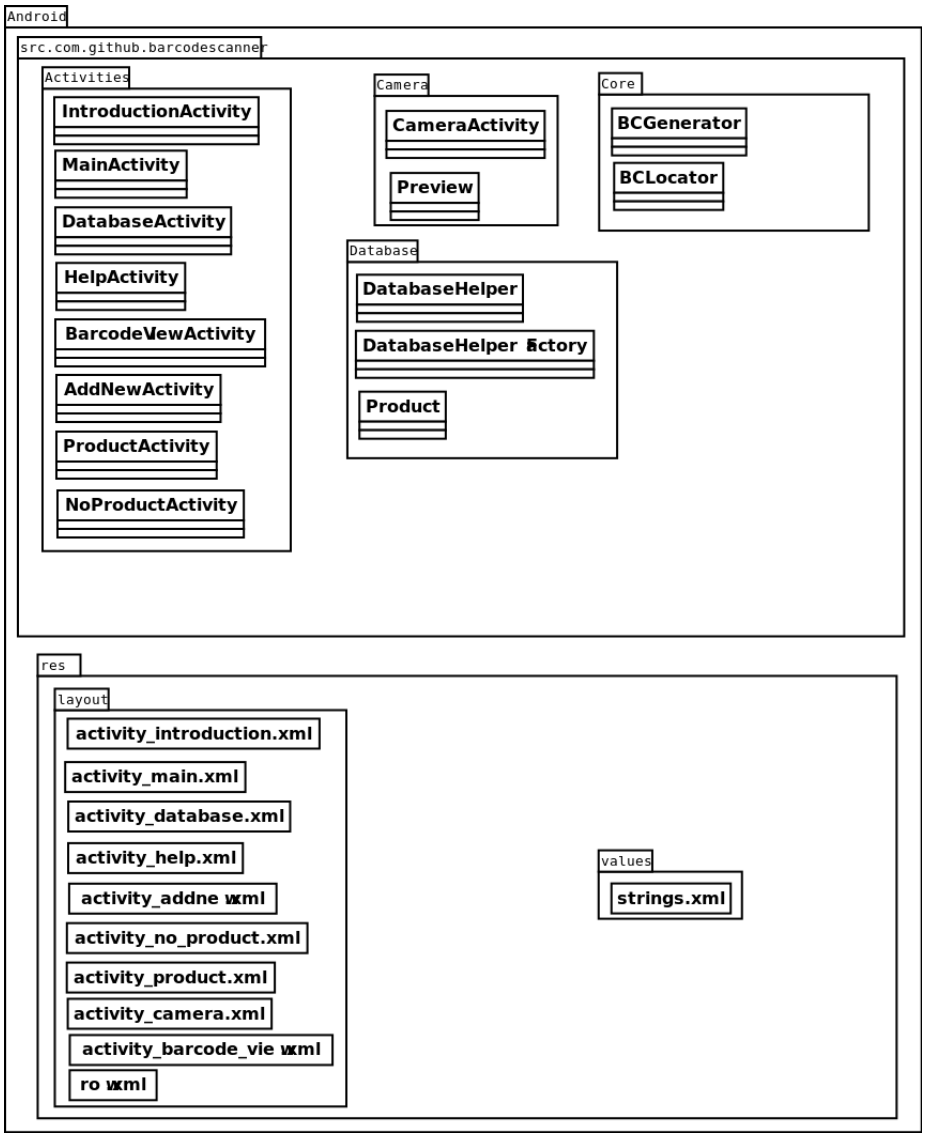

Figure 2.2: Package Diagram

2.3 Concurrency Issues

NA. The application is only tied to one mobile unit, meaning that only one instance of the application will run at once and each activity uses distinct, separate instructions for performing a task. The database is also only tied to one mobile unit. All together the application never performs any concurrent task that can interfere with another instruction in a harmful way.

2.4 Access control and security

NA

2.5 Boundary conditions

The application can only be run on mobile units running Android 4.0+.

Bibliography

[1] Mvc. http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller, May 2013.

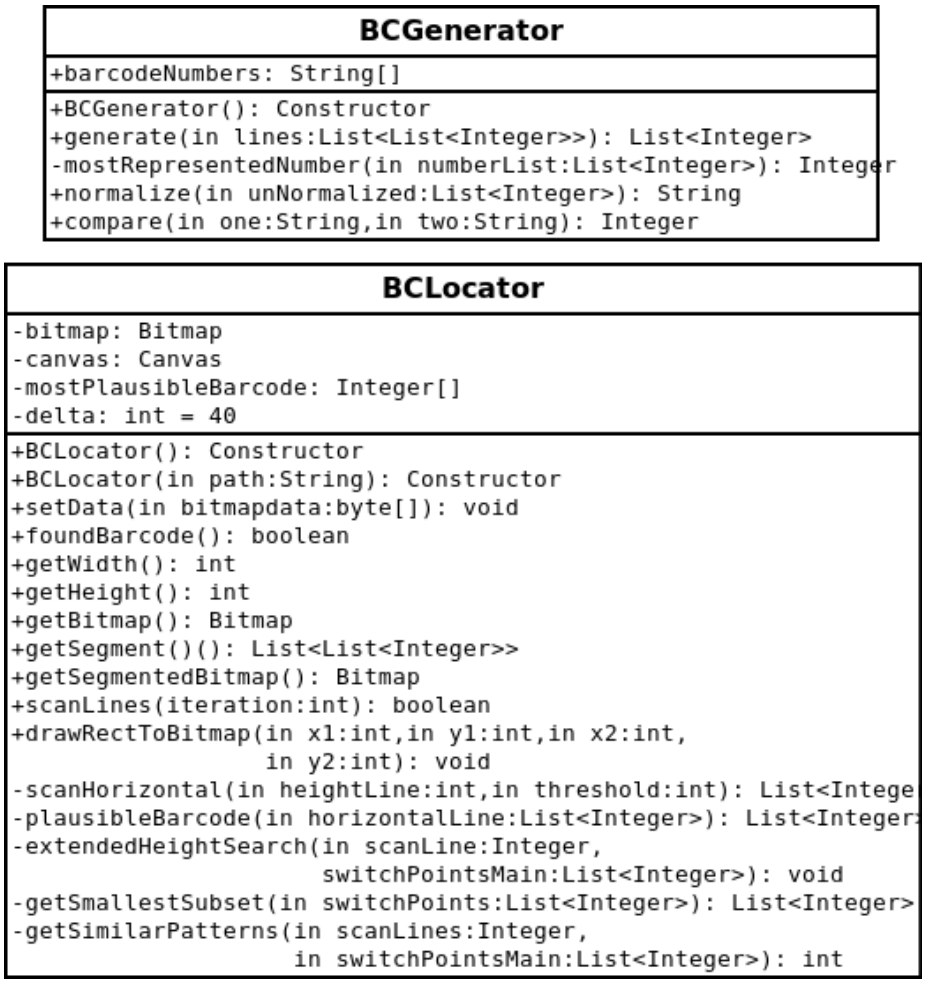

Figure 3: Class Diagram

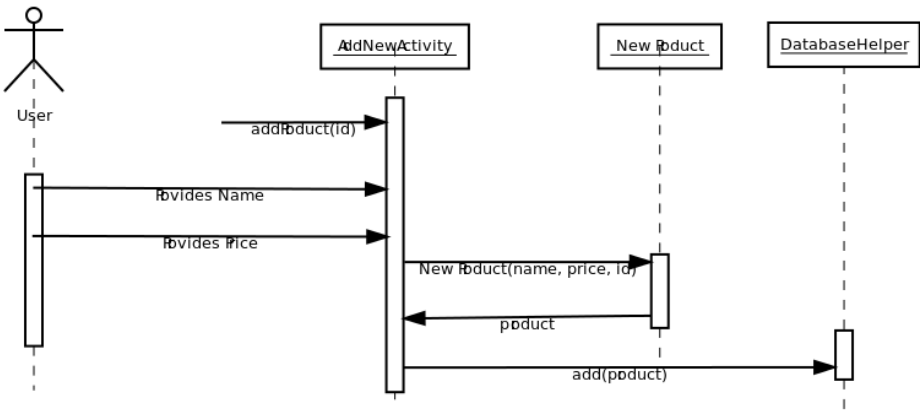

Figure 4: Sequence Diagram for adding a product to the database