Developer Manual

Lidköpings Stenhuggeri, 2013-09-27, v1

Lidkoping-SH applikation

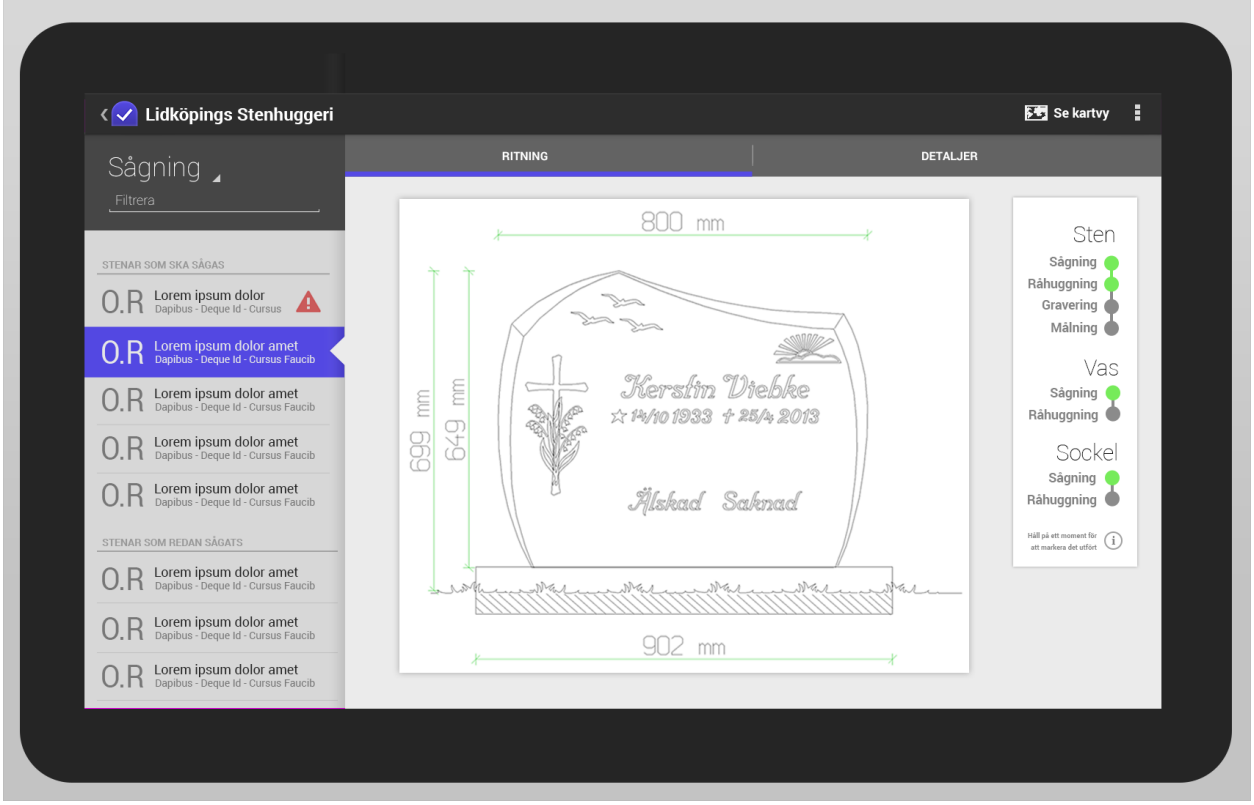

Table of Contents

Introduction

This document contains helpful information necessary for continuing the development of the application.

Overview

TODO: Import STAN images of completed application structure.

Build and install instructions

Requirements

Android SDK, including Android Development Toolkit (ADT) and Eclipse.
Everything you need to get started: http://developer.android.com/sdk/index.html

If you already have eclipse, use this link to download the Android Development Toolkit:
http://developer.android.com/tools/sdk/eclipse-adt.html

For more information about Android development, visit:
http://developer.android.com/tools/index.html

For better performance while running the Android emulator, if you are on an Intel processor system, install the Intel HAXM (Hardware Accelerated Execution Manager) extension through the Android SDK Manager.
http://software.intel.com/en-us/articles/installation-instructions-for-intel-hardware-accelerated-execution-manager-microsoft

Build

Clone the following project: https://github.com/Farenheight/Lidkoping-SH.git
You can clone with HTTPS, SSH, or Subversion.

Build project while in the master branch to use the latest version.

In order to continue development you need to be a member of the development group on GitHub and to register an SSH-key through your GitHub account (which corresponds to the one used in your Eclipse application).

Install

Install and run the .apk file on an android device or through the Android Emulator on a computer running Eclipse. You will find the .apk file in the /dist/ folder. It is recommended to use the latest version.

Application Design

Technologies

- Android Application written in Java together with Android for graphical user interface.
- Server application written in PHP with a MySQL database.
- SQLite database in Android.

Design Patterns

The application uses an independant model separated from the database, controllers and views. Views are static files in XML format, modified by controllers.

This design is somewhat closely related to the MVC (model-view-controller) pattern where the big difference is the android part, where the controller and view separation is in some ways diffuse.

Important Design Decisions

Static class

Using a static class to access the model. Easier with one class that can be used in all Android activities (controller) since the application is using the MVC pattern.

Database Layer

Using a DatabaseLayer between the model and the local database in order to keep Sql queries outside of the model.

Web service

A web service with a central database storage to synchronize data between all endpoints. The web service is written in PHP.

Server Layer

Using a ServerLayer between the internal modules and the remote server in order to keep logic handling http queries and JSon objects outside of the core.

Server - Client communication

While getting updates from server, we are sending an array, where each post contains another Long array containing two values:
1. Order's Identification Number.
2. Order's Timestamp.

These values are used to identify whether an order is going to be updated or not. Identifying the order by id, and if to be updated or not by the timestamp.

GIT - Branches

A new branch has been created for every new feature.

- **Docs**
  Containing document releases for the weekly releases.
- **Master**
  Main branch, to be merged into with every new feature branch when the feature implementation has been successfully completed.
- **Dev**
  Containing a new commit for each release, with a tag.