System design document for Hedgehog Photo
Version: 0.1
Date: 2012-04-27
Author: Hedgehog Photo
This version overrides all previous versions.

1 Introduction

1.1 Design goals
The design must be very modular and flexible, parts should be isolatable for ie testing. The core (model) should be separated from the view and the controller so that the application should easily be ported to other platforms like Android.

1.2 Definitions, acronyms and abbreviations

2 System design
The application will heavily rely on MVC to achieve modularity. The supplied plugins will also use MVC architecture. User written plugins can use any system architecture as long as they use Hedgehog Photos custom annotations.

2.1 Overview

2.1.1 Event handling
There will be no central event handling in our application. Every component will have its own unique controller.

2.1.X Plugin handling
Hedgehog Photo will have support for plugins. This makes the application very customisable and modular. Plugins will be loaded from a folder called plugin from the user home directory. This is will be customizable. Writing plugins should be an easy task and we want the devoloper to be in control as much as possible and thats why we have decided to have a very simple “API”. We aid plugin writing by only having two custom annotations one which initialises the plugin and one which gets the view. The plugin loader also compiles the plugins if its necessary (if the .class file is nonexistent or outdated).

2.1.X Stock plugins
Some stock plugins will be supplied with this applications. The stock plugins are: TagCloud, Map and Calendar. These plugins were made as they provide a richer experience while they at the same time were designed to stay to our core philosophy, simple and intuitive to use. These plugins are open for modifications. Users that want to write custom plugins can look at the source code to maybe get an even better understanding on how to write modifications to this program.

2.1.X Database handling

Hedgehog Photo will include a database. The database will save all necessary information-all information that will be shown in the program plus the paths- about the photo. You can get all information by using the DatabaseHandler class.

2.1.X Choose files

By using our file chooser(to be found in the menu) you can choose which files/directory that will be shown in Hedgehog Photo,

2.2 Software decomposition

2.2.1 General

Package diagram. For each package an UML class diagram in
appendix
The application is composed by the following modules, see figure (lägg in figur)

2.2.2 Layering

2.2.3 Dependency analysis

2.3 Concurrency issues

2.4 Persistent data management

2.5 Access control and security

2.6 Boundary conditions

3 References

APPENDIX