System design document for Paint++

Anthony Tao, Hampus Ekberg, Henrik Tao, August S'o'lveland

28/09-2018
1.0

1 Introduction

General info. What is this? What does it describe?

This is a system design document which describes the system and structure of Paint++.

1.1 Definitions, acronyms, and abbreviations

Some definitions etc. probably same as in RAD MVC - Model-View-Controller

2 System architecture

Paint++ is a raster graphics editor, which is a software program that allows the user to create art and edit images. The overall structure of the program follows the MVC pattern and only requires one computer. The MVC pattern is a software architecture pattern which divides the system into three main components, model, view and controller. The model is responsible for logic and data. The model does not have dependencies with the other components which are not a part of the model. The view is responsible for the graphics of the software, it updates the graphics using logic and data from the model component. The controller is responsible for processing events, such as user interactions, which can result in changes in both the model and view. Since it is a software application it starts by running the paint++ executable file and stops by exiting or closing down the application window.

2.1 Subsystem decomposition

The model of the system contains tool classes and interfaces connected to the tools. The view The controller

2.2 Model

The model component is responsible for managing data and logic. It is independent of the view, the user interface. In Paint++ the model contains the data and logic of the application and is divided into three subsystems which are model, tools and utilities. The model subsystem is responsible for the

The tool subsystem is responsible for the tools used in the application such as pencil, brush and fill tool. It contains all the tool classes along with specific methods of each tool. The utility subsystem contains utilities which are necessary for many tools, e.g. a PaintColor class which handles color.

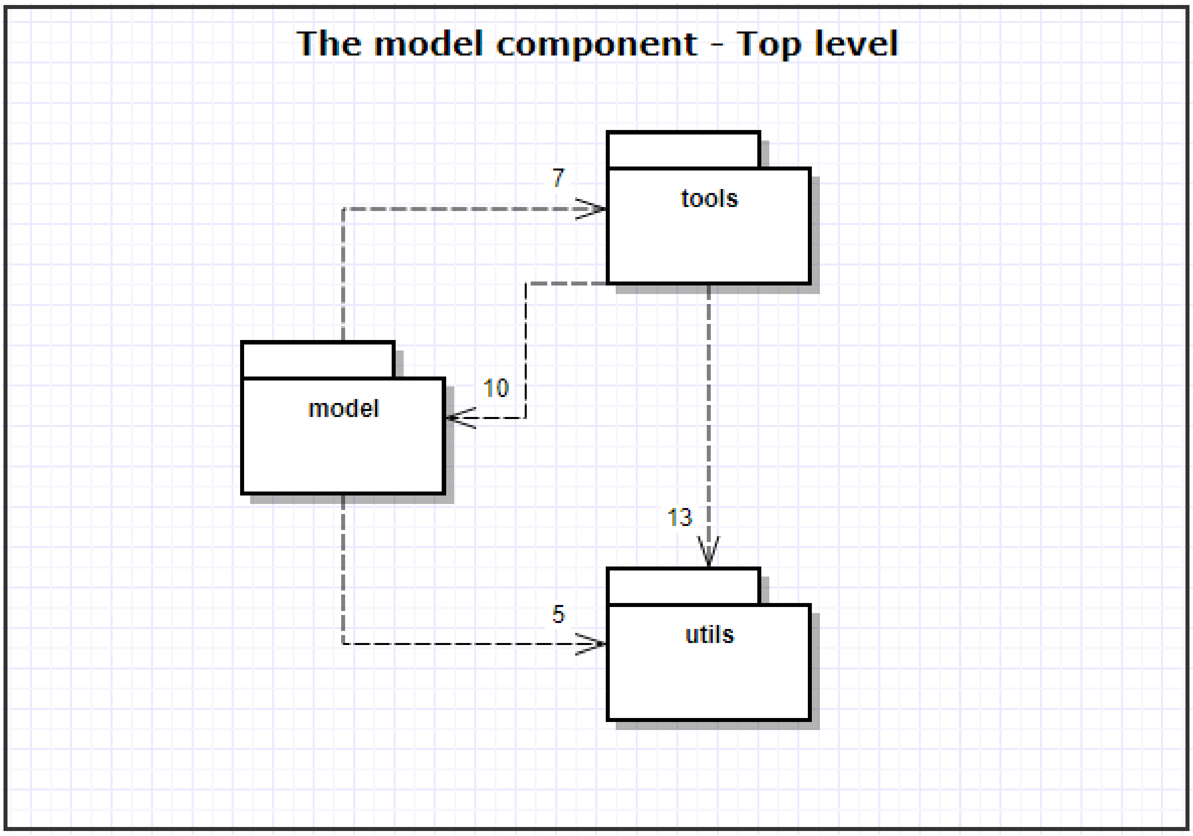


Figure 1: Model component top level

2.2.1 Model diagrams

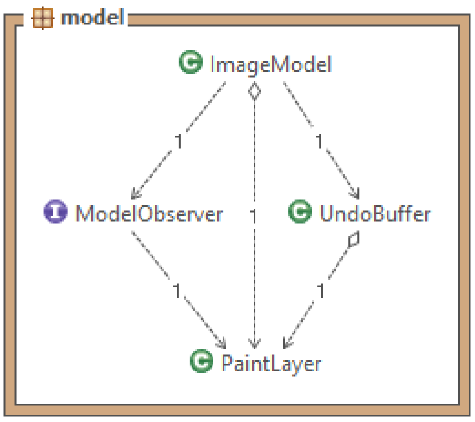


Figure 2: Model subsystem dependencies

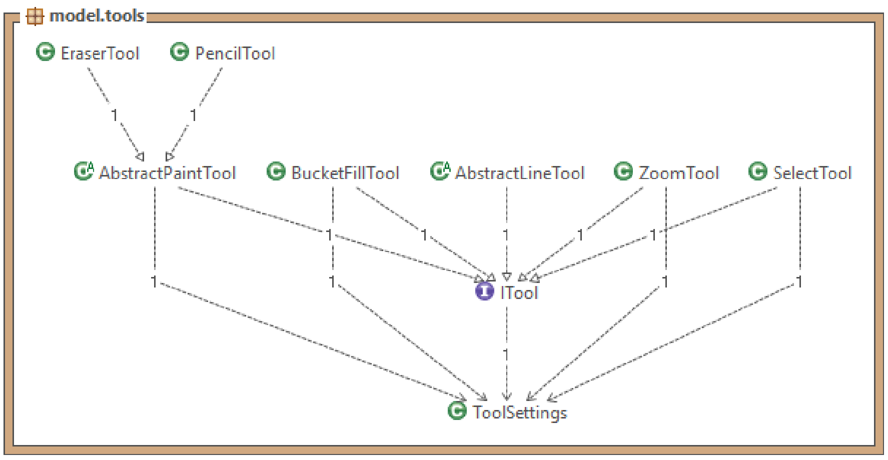


Figure 3: Tool subsystem dependencies

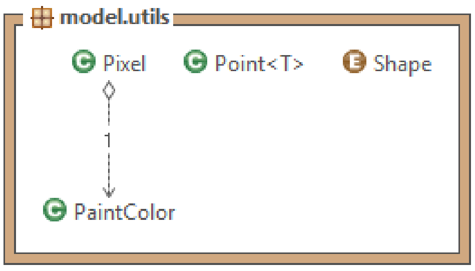


Figure 4: Utility subsystem dependencies

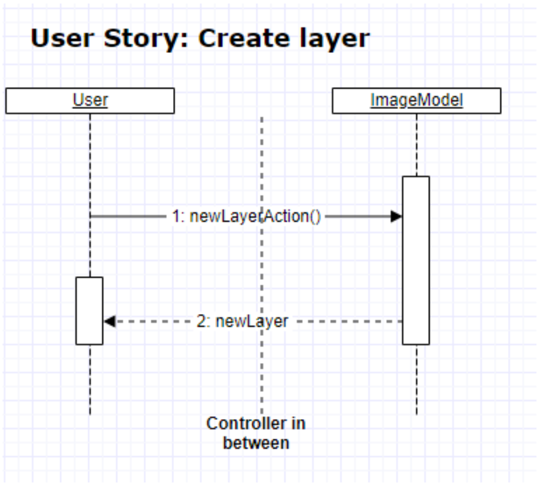


Figure 5: Flow for creating layers

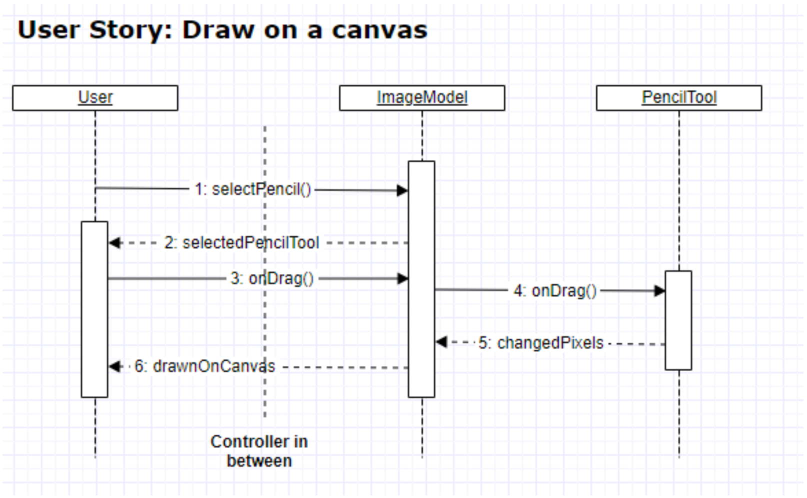


Figure 6: Flow for drawing on the canvas

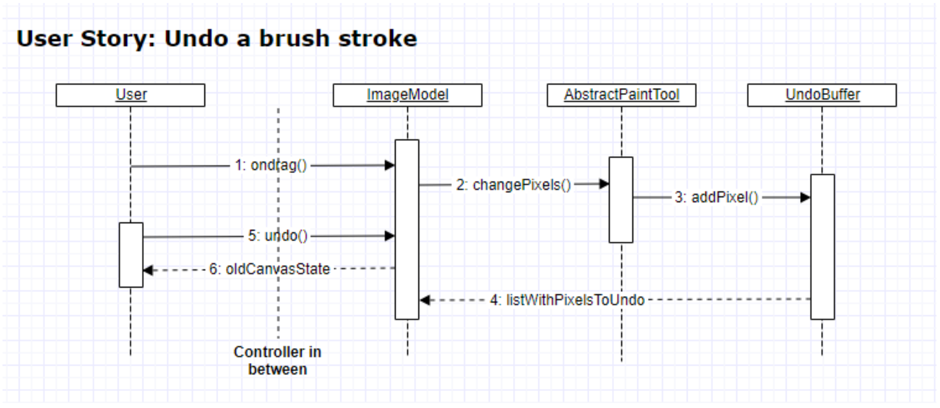
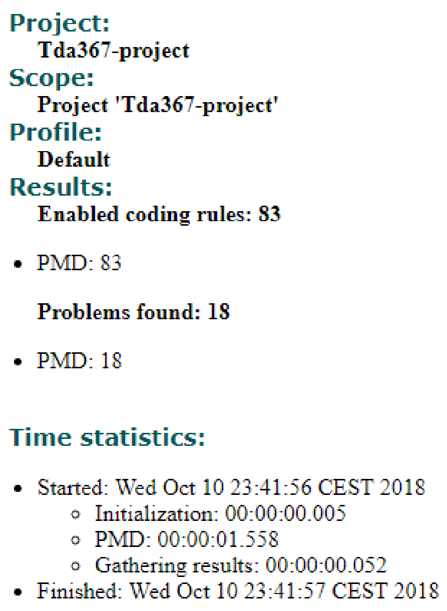


Figure 7: Flow for undoing brush stroke

2.2.2 Model quality



Project:
Tda367-project
Scope:
Project 'Tda367-project'
Profile:
Default
Results:
Enabled coding rules: 83

- PMD: 83

Problems found: 18

- PMD: 18

Time statistics:

- Started: Wed Oct 10 23:41:56 CEST 2018
  - Initialization: 00:00:00.005
  - PMD: 00:00:01.558
  - Gathering results: 00:00:00.052
- Finished: Wed Oct 10 23:41:57 CEST 2018

Figure 8: First PMD result

3 Persistent data management

How does the application store data (handle resources, icons, images, audio, ...). When? How? URLs, pathes, ... data formats... naming..

4 Access control and security

Paint++ is an application that can be used by anyone. There is no difference.
Different roles using the application (admin, user, ...)? How is this handled?

5 References