Image Album

Design Document: Version 3

22 April 2017

Contributors:

*Đivo Lise*

*Rachel Poturich*

*Mirko Šiljeg*

*Marko Živko*

Table of Contents

[Class Table 3](#_Toc478670546)

[Class Diagrams 3](#_Toc478670547)

[Overview 4](#_Toc478670548)

[Class Diagrams cont. 4](#_Toc478670549)

[Pattern Specific Class Diagrams 5](#_Toc478670550)

[Command Pattern 6](#_Toc478670551)

[Composite Pattern 6](#_Toc478670552)

[Pattern Rationale 7](#_Toc478670553)

[Command Pattern 8](#_Toc478670554)

[Composite Pattern 8](#_Toc478670555)

[Sequence Diagrams 9](#_Toc478670556)

[Add Label 9](#_Toc478670557)

[Crop Image 10](#_Toc478670558)

[Search 11](#_Toc478670559)

# 

# Class Table

|  |  |  |
| --- | --- | --- |
| **Class Name** | **Responsibilities** | **Collaborators** |
| ImageAlbum | Image album class will be responsible for creating overall GUI and running the application. | ViewMode |
| EditMode | Class that will be responsible for editing the images. For example, user will be able to rotate, resize or crop certain image. | Command Interface |
| ViewMode | Class to determine how the user interface should display and what operations should be available. | ImageAlbum  EditMode  Command Interface |
| ImgComponent | Abstract class to define Image and Album subclasses part of the Composite pattern | ViewMode  Command Interface |
| Command Interface | Interface for defining and executing commands | Concrete command classes |

# Class Diagrams

## Overview

# ../Desktop/Image%20UML%20-%20OVERVIEW%20(1).png

Figure – General system overview

Figure 1: This image is showing the general organization of classes. Also provided are relationships and all the interfaces that are needed for the command and composite patterns.

# Class Diagrams cont.

## ../Desktop/Image%20UML%20-%20display%20model%20(2).pngViewMode

# 

Figure 2 – ViewMode class diagram

Figure 2: This diagram shows the ViewMode class, which handles user interaction when the user is browsing images, searching images, or adding labels. The ViewMode can have a different mode, called EditMode, which displays elements needed for user interaction with image manipulation. ViewMode also has associated ImageCommands that refer to commands such as adding a label and searching.

# Pattern Specific Class Diagrams

## Command Pattern

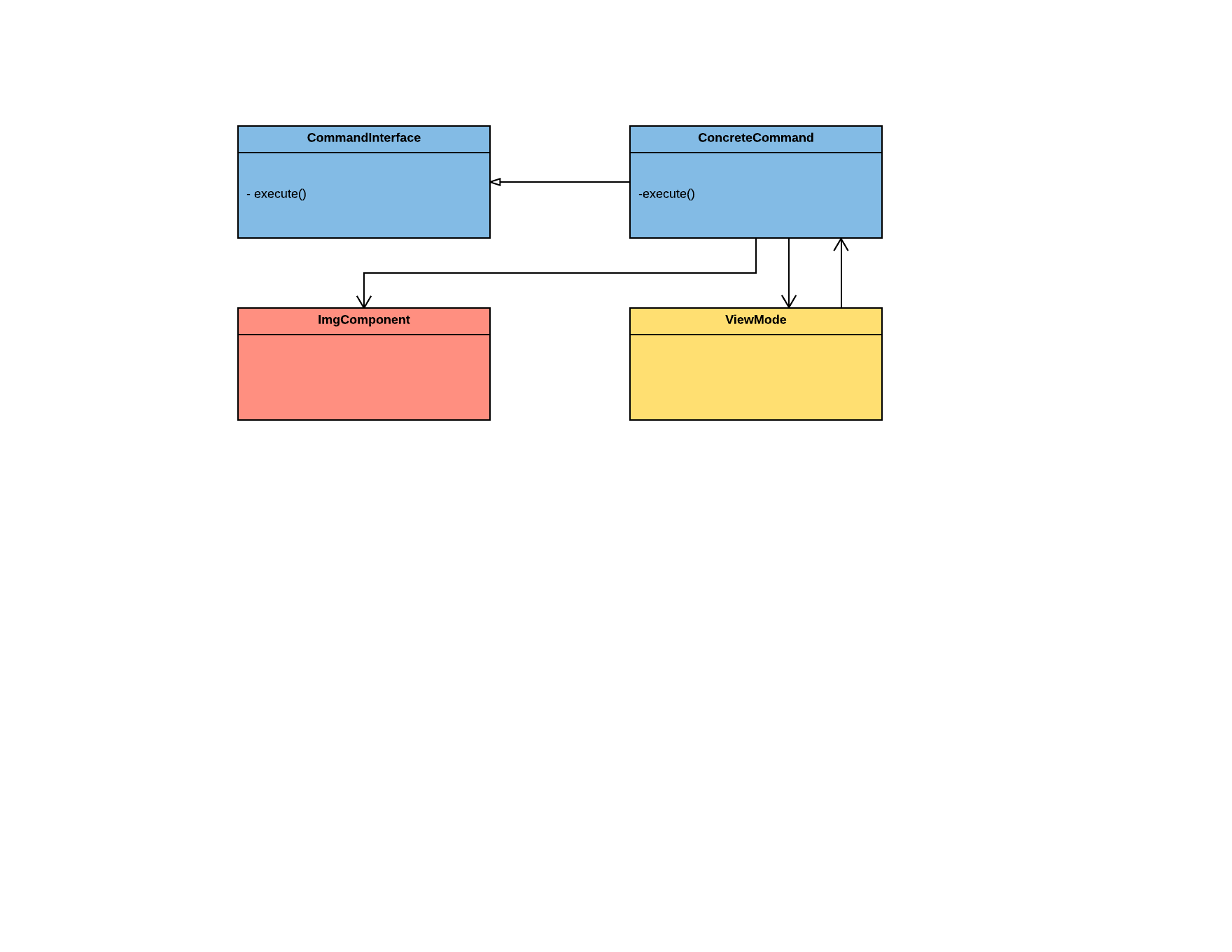


Figure 3 – The command pattern

## 

Figure 3: Command pattern will issue requests to objects without knowing anything about the receiver of the request or about how this operation is done. Receiver will have knowledge of what to do to handle the request, in our case that will be class called ViewMode. Invoker, in our case ImgComponent, will hold a command and will get a command to execute the request by calling the execute method.

## Composite Pattern

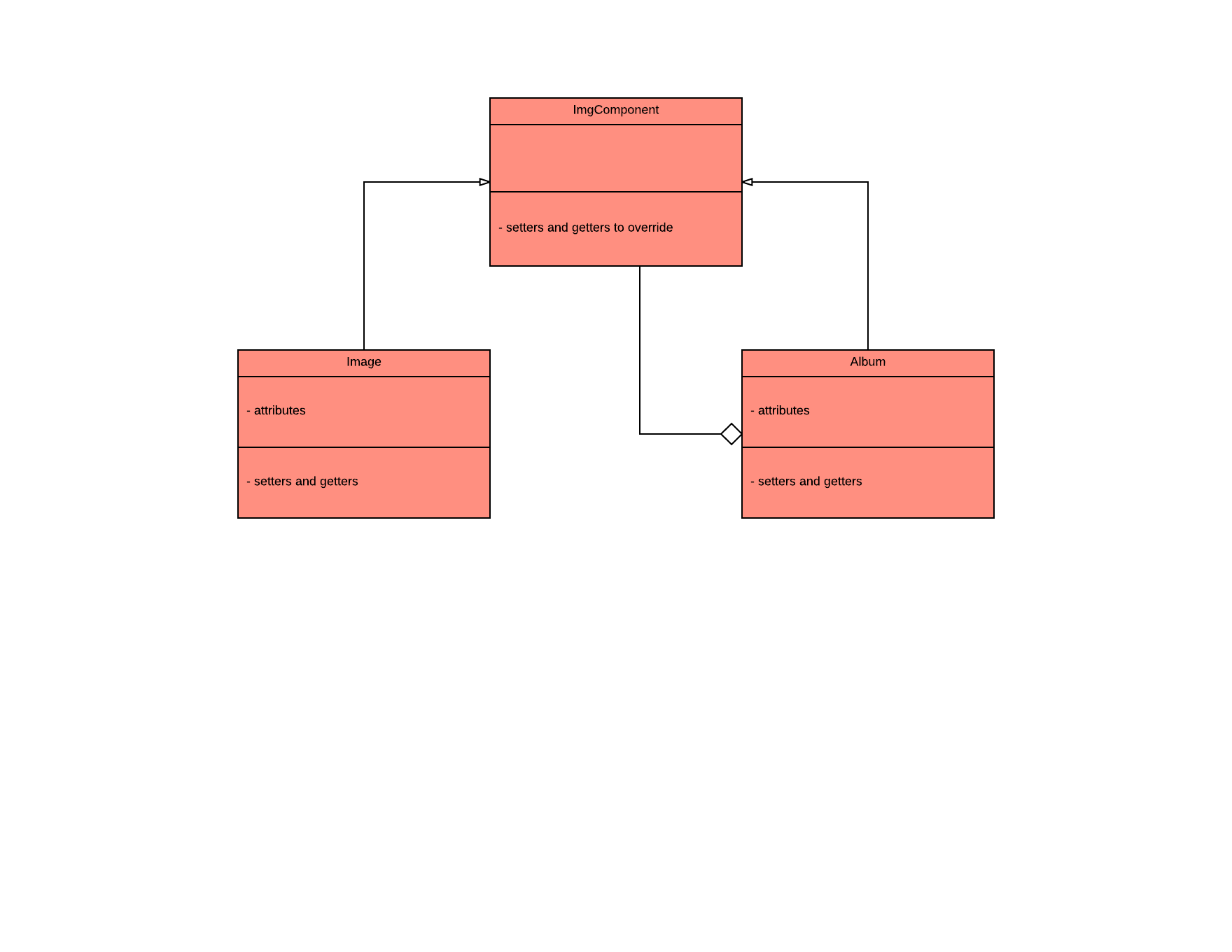


Figure 4 – The composite pattern with imgcomponents

Figure 4: ImgComponent will declare the abstract class for all the objects that are in the composition. It will also declare methods for managing child components. Album and Image will use ImgComponent class whose purpose will be to create Image and Album objects to be viewed in the program.

# Pattern Rationale

## Command Pattern

* The command pattern will be used in implementing the tools used during the EditMode of the program and the general ViewMode.

## Composite Pattern

* The Composite pattern will be used in implementing the Album-Image relationship

# 

# Sequence Diagrams

## ../Desktop/Sequence%20-%20Add%20Label%20-%20Page%201%20(3).pngAdd Label

Figure 5. Add Label to a Component

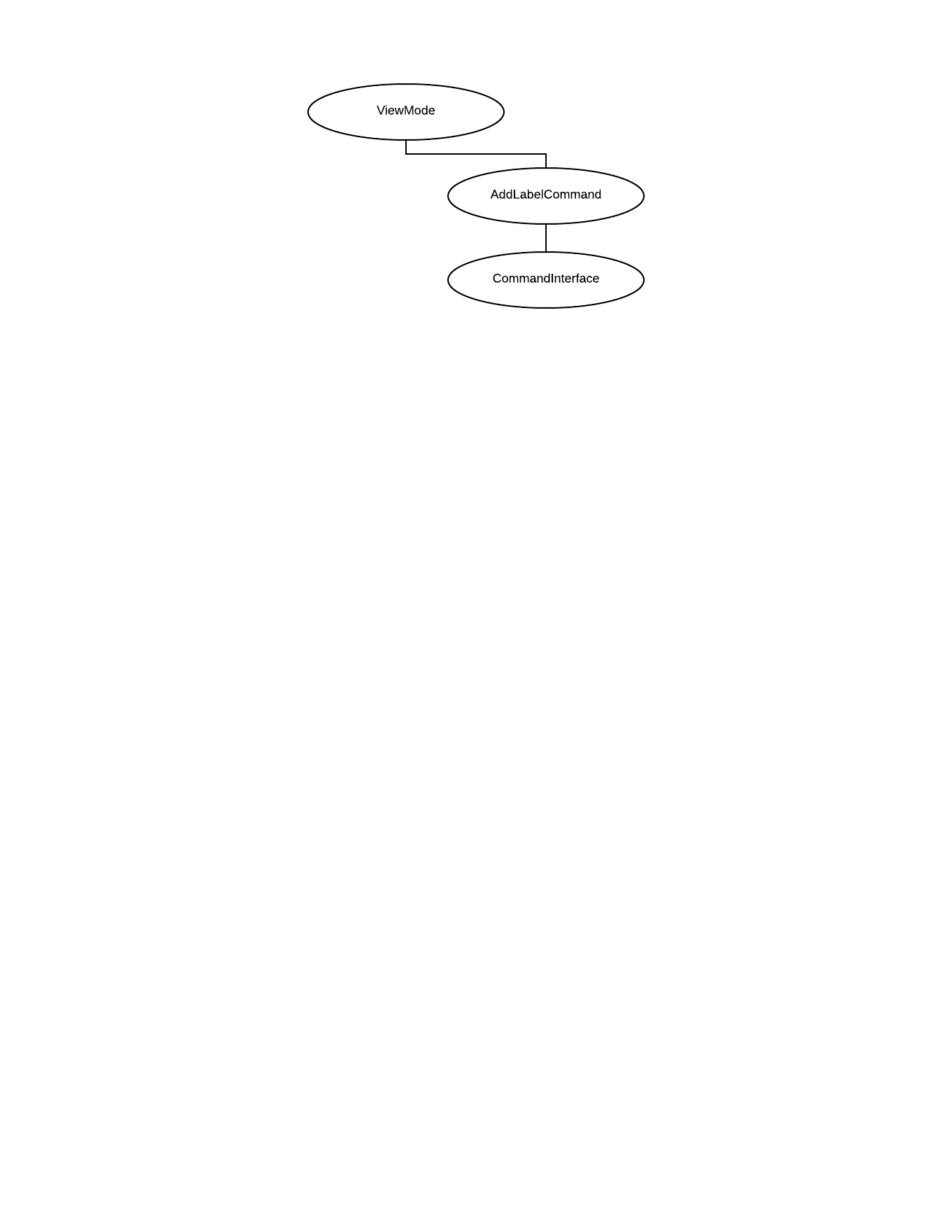


Figure . Object Diagram CORR. Fig. 5

## 

## ../Desktop/Sequence%20-%20Crop%20Image%20-%20Page%201%20(1).png Crop Image

Figure 7. Crop Image with a Command

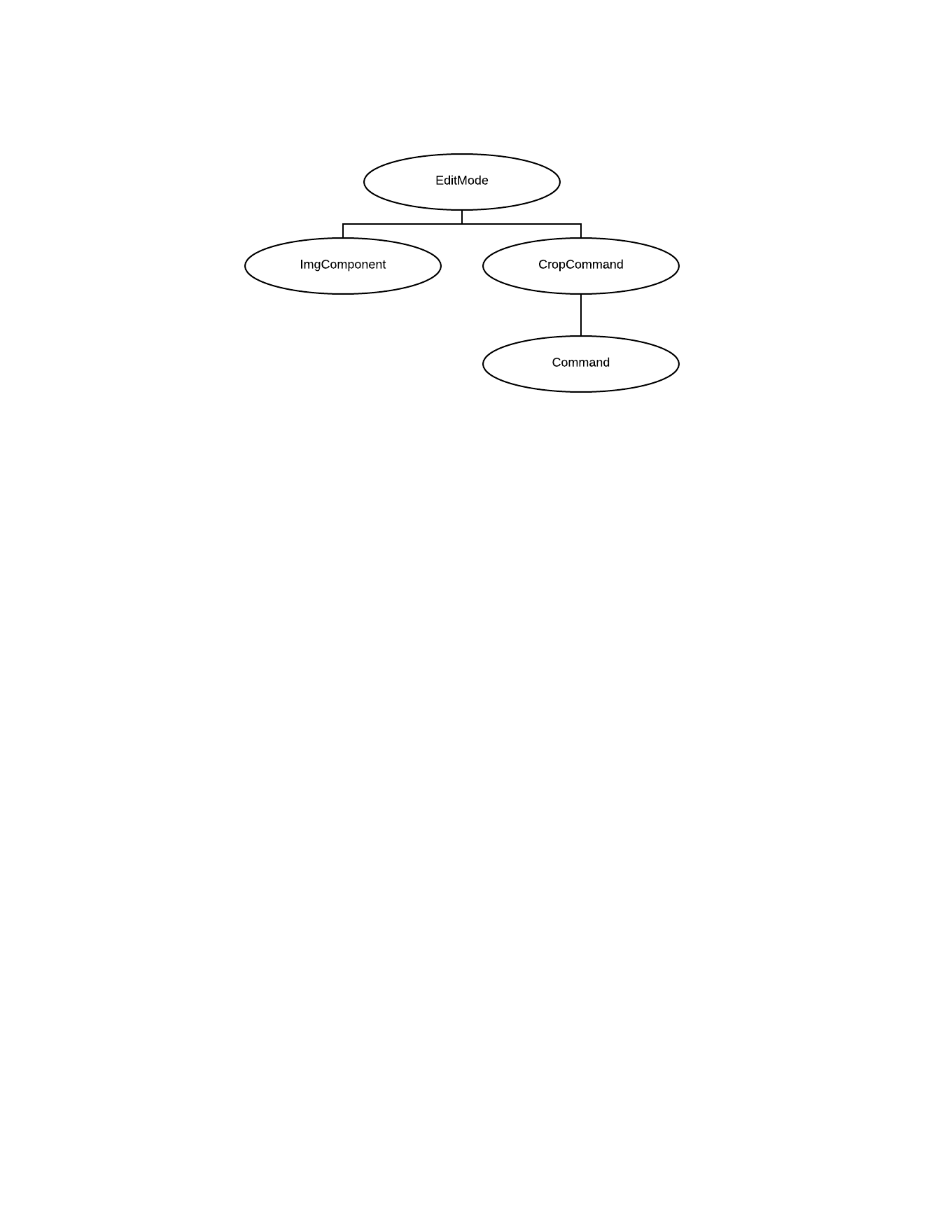


Figure . Object Diagram CORR. Fig. 7

## 

## **../Desktop/Sequence%20-%20Search%20-%20Page%201.png**Search

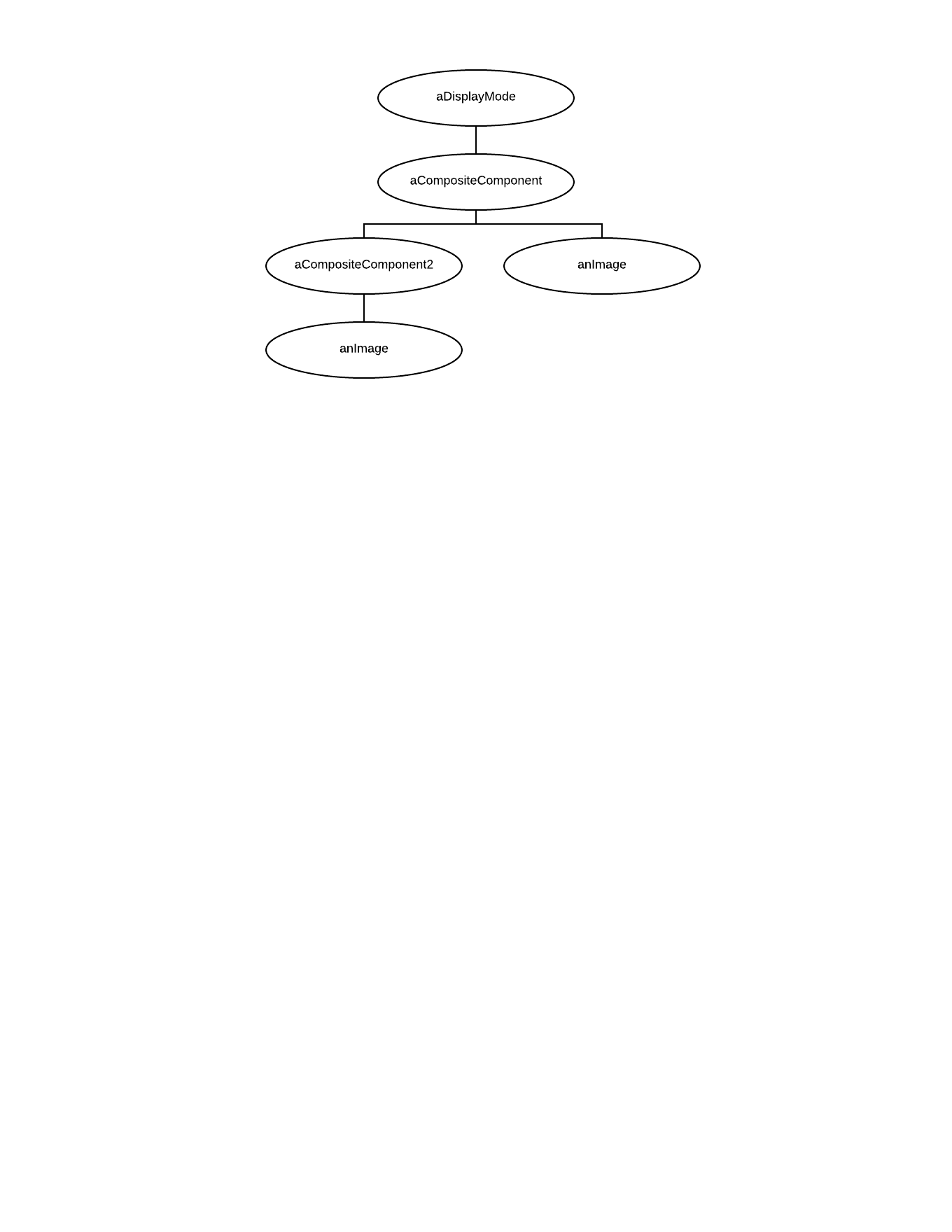
****

Figure . Object Diagram CORR. Fig. 9

Figure 9. Search for Components