YADIV Viewer

Designed And Developed By:

1. Ahmed Khalil Yakout(8)
2. Ahmed Mohamed Hamdy(14)
3. Rowan Mohamed AbdelMohsen(29)
4. Haya Mohamed AbcelMohsen(79)
5. Youssef Ahmed Darwish(85)

Introduction:

Image Viewer application are software implementations that ease display of image on screens and enables users to alter state of viewed image and perform simple edits that may include rotation and cropping.

YADIV Viewer Application:

YADIV is an image viewing application that implements the main functionalities on desktops as well as android platforms and was designed and developed to reach maximum satisfaction of its users.

Main Interface:

Main requirement of the image viewer application is displaying images suiting the majority of users. Several design decisions were taken to design the main interface of displaying images that is believed to achieve user satisfaction. Decisions such as Loading 1 image vs Loading multiple images, Fixing user interface options vs appearing on demand and auto resizing loaded images were all went through.

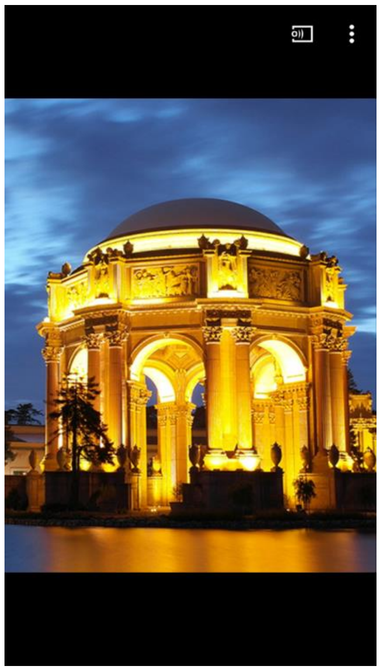


Fig1: Single Image Loaded on YADIV

One Image Vs Multiple Images: YADIV Applications load only a single image at a time the following designs aspects were considered.

	Loading one Image	Loading Multiple Images
Ease of Use	Simple	Complicated
Features Enabled	Less	More
Intended Output	Guaranteed	Maybe Ambiguous
Previous Implementations	Majority	Minority
Current User Appeal	Majority	Minority

Main Features Interface: YADIV allows user to reach any of its options instantaneously by docking them to application interface.

	Docking Features	On Demand
Access time	Instantaneous	Non-instantaneous
Area of Display (Android)	More Limited	Less Limited
Area of Display (Desktop)	Not Affected	Not Affected
Shape appeal (Desktop)	Favored	Disfavored
Shape appeal (Android)	Indifferent	Indifferent
Previous Implementations	Major in Desktop	Minor in Desktop
Previous Implementations	Minor in Android	Major in Android

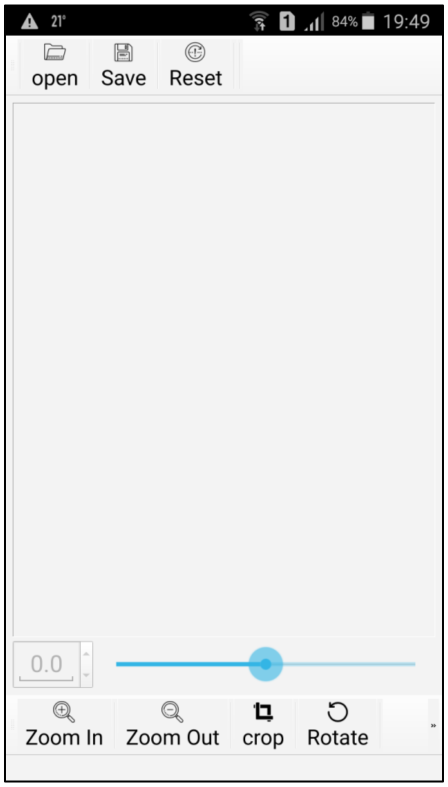


Fig2: Main Interface

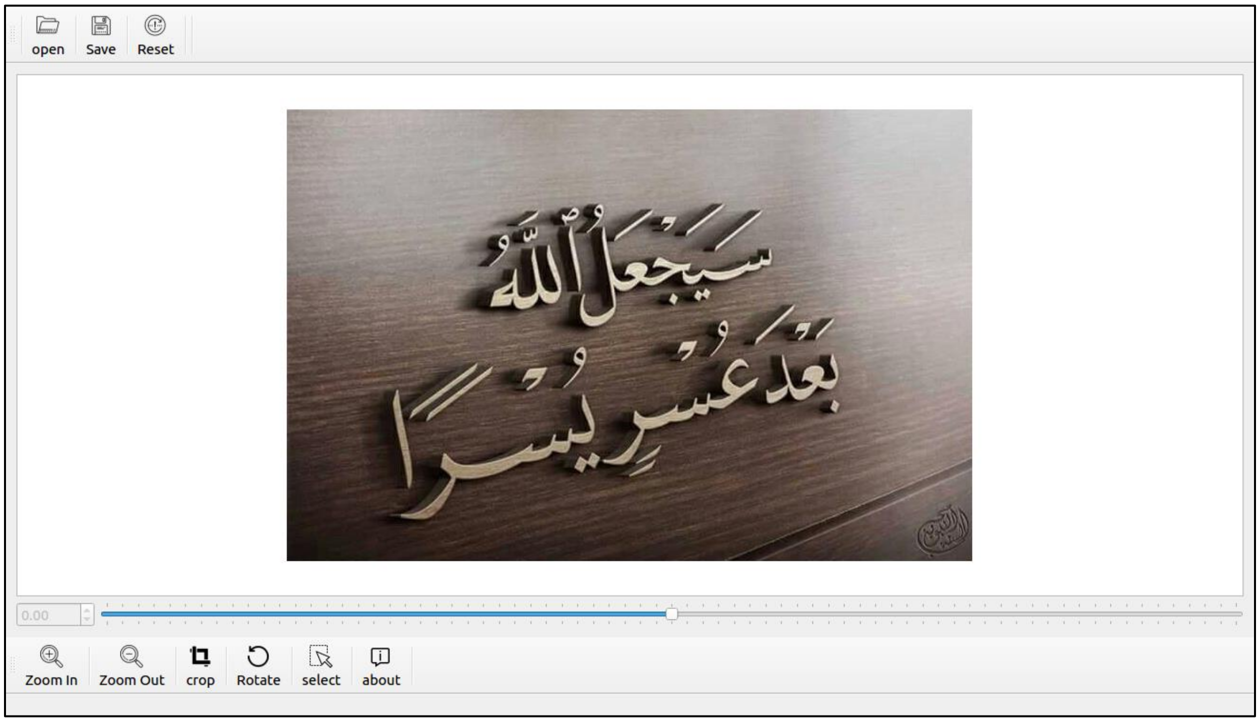


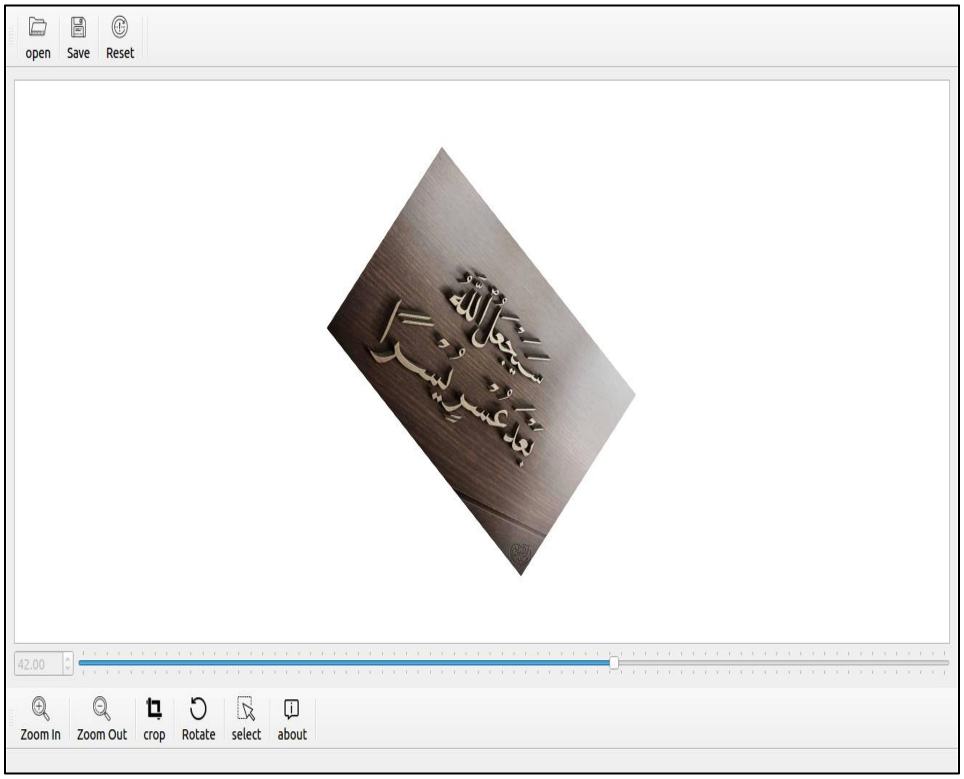
Fig3: Desktop Main Interface

The previous comparison shows great advantage of options docking in Desktop Application and a close tie between the two options in Android Application, In that case the easiest for development was considered and both applications implement options docking interfaces.

Main Features

- **Zoom:** Zooming in and out support both default image zooming and selected area zooming.
- **Crop:** crop feature supports selected area cropping and image replacement.
- **Rotation:** Rotation was implemented using linear on click slider to choose rotating angle, the image is rotated about it center.

	Linear Slider	Circular Gadget
Process time	Fast	Fast
Error sensitivity	Unlimited	Limited
Size in display	Occupies full of Width	Area efficient
Shape appeal (Desktop)	Favored	Disfavored
Shape appeal (Android)	Disfavored	Favored
Mouse Access	Highly Favored	Poorly Recommended
Touch Access	Favored	Less Favored



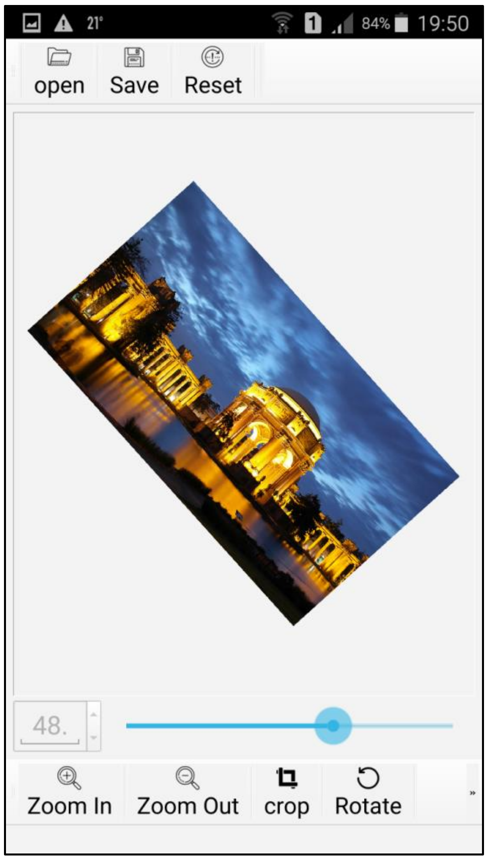


Fig4: Rotation interfaces

Area Selection: Area selection is user friendly using mouse or touchpad swiping.

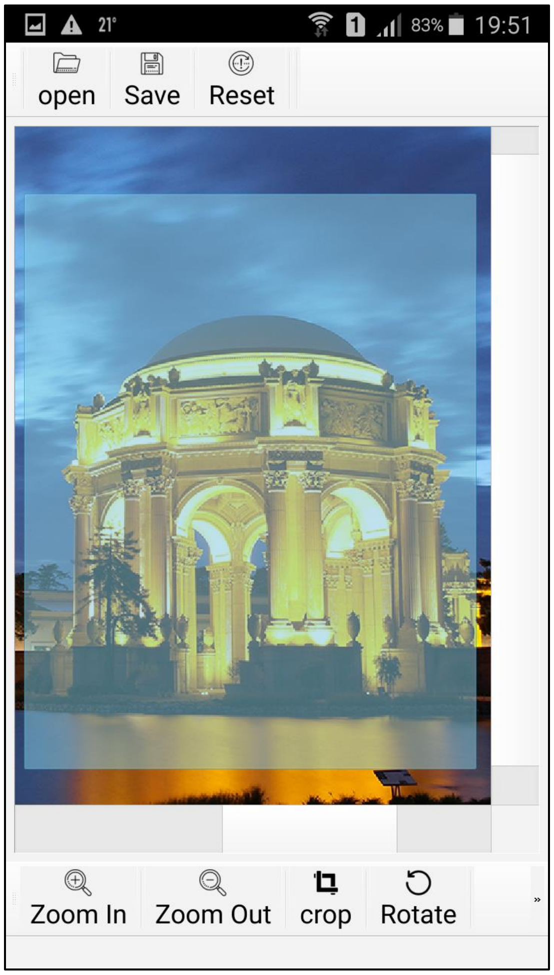


Fig5: Selection Area

Software Rationale:

Our software was implemented to achieve simplicity and enables developers to easily plug in new features and interfaces. The applications was developed upon QT platform in C++ and XML and was implemented in 2 Main layers :- Interface Layer (coded in XML and C++) and Control Layer (coded in C++) Each is divided in sublayers implemented in separate classes. The main aim of having a layered software architecture was following the QT platform plug ins and having a debug-able and extendible architecture.

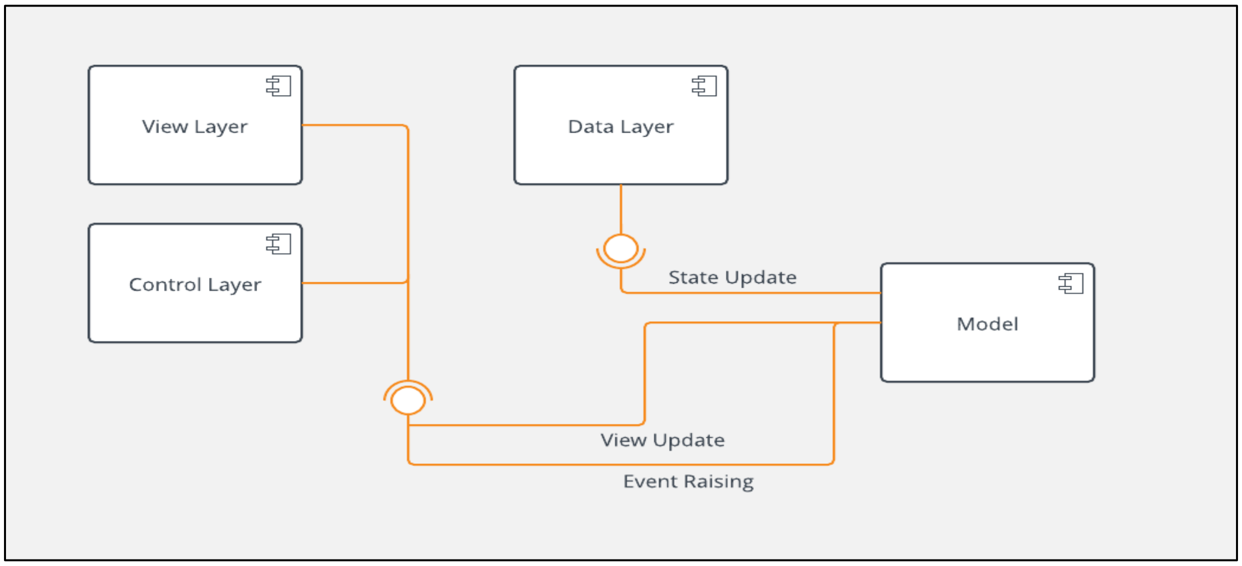


Fig6: Software Design

Main Classes

- **ImageViewer Class:** Main class implementing Model layer that listens to events raised by interface and basic functionalities are implemented in the class.
  Classes associated: Image , GraphicViewn , Qpixmap

- **GraphicViewn Class:** Class implementing Control Layer responsible for augmenting the events raised and selection area display.
  Classes associated : QRubberBand , QMouseEvents , QTouchEvents

- **Image Class:** The class main acts as a data-holder for displayed image data as rotation angle , zoom scale and image raw data and pixmap representation and is reset upon loading and undo.
  Classes associated : Qpixmap , QImage

Task	**Assigned on**	**Assigned to**	**Due**	**Finished**
Qt Installation	25/10	Team	27/10	27/10
Requirements Analysis	25/10	Team	25/10	25/10
Interface Analysis	25/10	Team	27/10	27/10
Interface Design	25/10	Youssef	27/10	27/10
Zoom In/Out	25/10	Ahmed Yakout	1/11	1/11
Rotate	25/10	Ahmed Yakout	1/11	1/11
Selection	25/10	Ahmed Hamdy	1/11	1/11
Crop	25/10	Haya	1/11	1/11
Save	25/10	Rowan	1/11	1/11
Zoom to area	10/11	Haya , Rowan	13/11	13/11

Touch Zoom (Android)	10/11	Yakout,Youssef	13/11	12/11
Touch Select (Android)	10/11	Hamdy	13/11	13/11
Android Save	13/11	Haya , Rowan	14/11	14/11
Documentation	14/11	Team	14/11	14/11