Hazur

Submitted for partial fulfilment of the Degree
of
Bachelor of Technology
(Computer Science and Engineering)

Submitted By:
Amitoj Singh
1410955
135377

Submitted To:
Sukhjit Singh Sehra
Training Coordinator
CSE Department

Department of Computer Science & Engineering
Guru Nanak Dev Engineering College
Ludhiana 141006

Acknowledgement

I, student of Guru Nanak Dev Engineering College, Ludhiana, have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

The author is highly grateful to Dr. M.S. Saini Director, Guru Nanak Dev Engineering College, Ludhiana for providing him with the opportunity to carry out his Six Weeks Training at Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana.

The author would like to whole heartedly thank Mr. Inderpreet Singh Founder of Ishwerdas Softwares who is a vast sea of knowledge and without whose constant and never ending support and motivation, it would never have been possible to complete the project and other assignments so efficiently and effectively.

I also want to thanks Gauravjeet Singh(D3CSE), Khushwinder kaur(D3IT) and all other trainees. Without their encouragement, it would not have been possible to complete this project in such an efficient manner.

Amitoj Singh

Abstract

CAD development project discuss the work done in computer-aided-design. Computer-aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design. I explored LibreCAD Source Code. LibreCAD is Free and Open Source CAD Software. LibreCAD is a fully comprehensive 2D CAD application that you can download and install for free. There is a large base of satisfied LibreCAD users worldwide, and it is available in more than 20 languages and for all major operating systems, including Microsoft Windows, Mac OS X and Linux . Librecad is an application for computer aided design in two dimensions . With librecad you can create technical drawings such as plans for buildings, interiors, mechanical parts or schematics and diagrams. Basically, LibreCAD is used to make 2D design.

This project also includes work regarding qt creater to make a text editor. Qt Creator is a cross-platform C++, JavaScript and QML integrated development environment which is part of the SDK for the Qt GUI Application development framework.

Furthermore, this project involves hatching technique implementation using C++ and Cairo graphics library. Hatching is the technique to fill any closed contour with different patterns or colors and is used by many CAD softwares.

Also, this project is completely open source and the entire code is available to the user as and when required. There is also Complete developer's Documentation as well as User manual alongwith it that helps using it a lot easier.

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION TO ORGANISATION

1.1 Ishwerdas Softwares

I had my Six Weeks Industrial Training at Ishwerdas Software Samrala . Ishwerdas is a small design, branding and software company that believes in changing the way the world works. With it's humble origins in a small town names Samrala, Ludhiana It's a start-up that aims to change the system by providing excellent services and creating the the most intuitive, user friendly software on web, desktop, mobile, watches and so on.

Ishwerdas Softwares is set up to make a difference at 3 fronts.

Firstly, Ishwerdas softwares is working hard to bring a change in education. Ishwerdas softwares believes in the ideas of Sir Ken Robinson and many like him who say that education system has been going into wrong direction for almost a century now. Three inherent properties of humans a.k.a "Diversity - Each human is inherently different", "Curiosity - Each human (child) is born curious", "Creative - Each child is a born artist" are being attacked by education system under the names of "Compliance", "Conformity" and "Discipline". Ishwerdas software believes that teaching happen, when learning happens and if there's no learning, there's no education and poor teaching. Ishwerdas softwares thus believe that a student never fails to learn until teacher fails to teach. Ishwerdas Softwares is trying to cure education system of it's disease by taking baby steps toward providing an alternative education system. First step being change in Industrial training. We have adapted ideas from various schools of education such as Shaolin education system, Ancient Indian Education System (Gurukuls) , improving over them using latest technology and incorporating it in our daily teachings. Ishwerdas softwares will also soon be launching software and technology to enhance the basic pre-nursery education. We are having talks with various schools to bring a change in the way teaching happens. We are set out to eliminate grades, marks and comparison among students starting from pre-nursery level. If you have watched 3-Idiots , you can exactly imagine the kinds of schools that we are trying to build.

Whatever else we do, we do it to support our major cause. Secondly, we see that many and most of Indian softwares, especially in enterprise suffer from bad taste. They have a bad User Interface, a tasteless User Experience and wastes the time of lac of users and no one seems to care

about those. So we as a group of developers and designers feel obligated to provide services in such a manner that each software is built at back-end in such a way that it loads fast, saves user's time and in front-end have an excellent and intuitive, emotion invoking interface. Our Industry needs to understand that users are human, and humans are driven by emotions (whether they admit or not). So emotionally intelligent interfaces can not only drive sales high but also provide user a relief and ease. No human deserves a crappy interface with 100 input fields and never ending drop downs.

Last but not the least, Ishwerdas softwares is set out to be an example of flat organisation structure. Yes, we agree with benefits of hierarchy and vertical ladder of growth but our managers would rather like to challenge status quo and work with likes of 37 signals, valve and github to create an organisation with flat structure. Ishwerdas is set out to lead by example in this area by becoming one of the very few companies in India by adopting a completely flat organisation structure.

Ishwerdas softwares is yet small but ambitious, and is fearless in choosing the road less traveled by.

CHAPTER 2

INTRODUCTION TO PROJECT

2.1 Introduction

The term computer graphics includes almost everything on computers that is not text or sound. Today almost every computer can do some graphics, and people have even come to expect to control their computer through icons and pictures rather than just by typing. Here in our lab at the Program of Computer Graphics, we think of computer graphics as drawing pictures on computers, also called rendering. The pictures can be photographs, drawings, movies, or simulations pictures of things which do not yet exist and maybe could never exist. Or they may be pictures from places we cannot see directly, such as medical images from inside your body. Computer graphics is now used in various fields; for industrial, educational, medical and entertainment purposes. The aim of computer graphics is to visualize real objects and imaginary or other abstract items. In order to visualize various things, many technologies are necessary and they are mainly divided into two types in computer graphics: modeling and rendering technologies.

2.2 Introduction to CAD

Computer-aided design (CAD) is the use of computer systems to assist in the creation, modification, analysis, or optimization of a design. CAD software is used to increase the productivity of the designer, improve the quality of design, improve communications through documentation, and to create a database for manufacturing.CAD is an important industrial art extensively used in many applications, including automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetics, and many more. CAD is also widely used to produce computer animation for special effects in movies, advertising and technical manuals. CAD output is often in the form of electronic files for print, machining, or other manufacturing operations. CAD is also used for the accurate creation of photo simulations that are often required in the preparation of Environmental Impact Reports, in which computer-aided designs of intended buildings are superimposed into photographs of existing environments to represent what that locale will be like were the proposed facilities allowed to be built. Computer Aided Drafting describes the process of creating a technical drawings with the use of computer software. CAD software is used to increase the productivity of the designer, improve the quality of design, improve communications

through documentation, and to create a database for manufacturing. CAD output is often in the form of electronic files for print or machining operations. CAD software uses either vector based graphics to depict the objects of traditional drafting, or may also produce raster graphics showing the overall appearance of designed objects.

Today there are very few aspects of our lives not affected by computers. Practically every cash or monetary transaction that takes place daily involves a computer. In many cases, the same is true of computer graphics. Whether you see them on television, in newspapers, in weather reports or while at the doctors surgery, computer images are all around you. A picture is worth a thousand words is a well known saying and highlights the advantages and benefits of the visual ation of our data. We are able to obtain a comprehensive overall view of our data and also study features and areas of particular interest. A range of tools and facilities are available to enable users to visualize their data, and this document provides a brief summary and overview. Computer graphics can be used in many disciplines. Charting, ations, Drawing, Painting and Design, Image Processing and Scientific Visualization are some among them. Computer graphics is concerned with all aspects of producing images using a computer. It concerns with the pictorial synthesis of real or imaginary objects from their computerbased models.

CAD often involves more than just shapes. As in the manual drafting of Technical and Engineering Drawings, the output of CAD must convey information, such as material, processes,dimensions and tolerances, according to applicationspecific conventions. CAD may be used to design curves and figures in twoDimensional(2D) space; or curves, surfaces, and solids in three dimensional(3D) space.CAD is an important industrial art extensively used in many applications, including automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetic, and many more. CAD is also widely used to produce Computer animation for special Effects in movies,advertising and technical manuals. The modern ubiquity and power of computers means that even perfume bottles and shampoo dispensers are designed using techniques unheard of by engineers of the 1960s. Because of its enormous economic importance, CAD has been a major driving force for research in computational geometry, computer graphics (both hardware and software), and discrete differential geometry.The design of geometric model for object shapes, in particular, is occasionally called Computer Aided Geometric Design (CAGD).While the goal of automated CAD systems is to increase efficiency, they are not necessarily the best way to allow newcomers to understand the geometrical principles of Solid Modeling.

I explored LibreCAD's source code. LibreCAD is Free and Open Source CAD Software. LibreCAD is a fully comprehensive 2D CAD application that you can download and install for free. There is a large base of satisfied LibreCAD users worldwide, and it is available in more than 20 languages and for all major operating systems, including Microsoft Windows, Mac OS X and Linux (Debian, Ubuntu, Fedora, Mandriva, Suse ...). Librecad is an application for Computer Aided Design (CAD) in two dimensions (2d). with LibreCAD you can create technical drawings such as plans for buildings, interiors, mechanical parts or schematics and diagrams.

The app is great for industrial designers, but anyone who wants to learn how to make 2D CAD drawings will like this program. For a free software, LibreCAD gives you a lot of tools to work with. New users will be able to create basic drawings, while advanced users can make engineering plans with 5the software. Layers can be added, ideal for complex drawings. The provided tools are sufficient for producing high precision drawings. You can start drawings from scratch. But it is

also easy to put in splines, ellipses, arcs, lines and circles. A single item can have several iterations. For instance, you have 4 modes for a rectangle parameter. The different shapes can be combined easily. LibreCAD also has a powerful zoom tool that lets you look at models at different distances. This is essential for designers who are going to make lifesize copies of a drawing. There are three tabs above the working area. The first tab is for changing color, useful for layer definition. The other tab is for changing size and the third for workspace customization.

LibreCAD also has grids which are extremely useful for those new to CAD. Once you have made the basic object, you can customize it in many ways. Scaling is particularly easy here. Also worth mentioning here is the "Explode text into letters" effect. It is a special feature that will come in handy ations. LibreCAD allows you to put horizontal or vertical restrictions on completed models. Relative zeros may be locked, useful for ending and starting points. All in all, it is powerful, free CAD application. You can download, install and distribute LibreCAD freely, with no fear of copyright infringement.

2.2.1 Introduction to LibreCAD

LibreCAD is a fully comprehensive 2D CAD application that you can download and install for free. There is a large base of satisfied LibreCAD users worldwide, and it is available in more than 20 languages and for all major operating systems, including Microsoft Windows, Mac OS X and Linux (Debian, Ubuntu, Fedora, Mandriva, Suse ...). Librecad is an application for computer aided design (cad) in two dimensions (2d). with librecad you can create technical drawings such as plans for buildings, interiors, mechanical parts or schematics and diagrams.

The app is great for industrial designers, but anyone who wants to learn how to make 2D CAD drawings will like this program. For a free software, LibreCAD gives you a lot of tools to work with. New users will be able to create basic drawings, while advanced users can make engineering plans with the software. Layers can be added, ideal for complex drawings. The provided tools are sufficient for producing high precision drawings. You can start drawings from scratch. But it is also easy to put in splines, ellipses, arcs, lines and circles. A single item can have several iterations. For instance, you have 4 modes for a rectangle parameter.

The different shapes can be combined easily. LibreCAD also has a powerful zoom tool that lets you look at models at different distances. This is essential for designers who are going to make lifesize copies of a drawing. There are three tabs above the working area. The first tab is for changing color, useful for layer definition. The other tab is for changing size and the third for workspace customization. LibreCAD also has grids which are extremely useful for those new to CAD. Once you have made the basic object, you can customize it in many ways. Scaling is particularly easy here. Also worth mentioning here is the "Explode text into letters" effect. It is a special feature that will come in handy ations. LibreCAD allows you to put horizontal or vertical restrictions on completed models. Relative zeros may be locked, useful for ending and starting points. All in all, it is powerful, free CAD application. You can download, install and distribute LibreCAD freely, with no fear of copyright infringement.

2.2.2 LibreCAD's features

- It's free no worry about license costs or annual fees.
- No language barriers it's available in a large number of languages, with more being added continually.
- GPLv2 public license you can use it, customize it, hack it and copy it with free user support 15and developer support from our active worldwide community and our experienced developer team.
- LibreCAD is an Open Source communitydriven project: development is open to new talent and new ideas, and our software is tested and used daily by a large and devoted user community; you, too, can get involved and influence its future development.
- LibreCAD is an Application for Computer Aided Design (CAD) in two dimension (2D). With LibreCAD you can create technical drawings such as plans for building, interiors, mechanical parts or schematics and diagrams.

2.2.3 How it started?

LibreCAD started as a project to build CAM capabilities into the community version of QCad for use with a Mechmate CNC router. LibreCAD is a version of QCad CE ported to Qt4. Since QCad CE was built around the outdated Qt3 library, it had to be ported to Qt4 before additional enhancements. This gave rise to CADuntu.

The project was known as CADuntu only for a couple of months before the community decided that the name was inappropriate. After some discussion within the community and research on existing names, CADuntu was renamed to LibreCAD.

Porting the rendering engine to Qt4 proved to be a large task, so LibreCAD initially still depended on the Qt3 support library. The Qt4 porting was completed eventually during the development of 2.0.0 series, thanks to our master developer Rallaz, and LibreCAD has become Qt3 free except in the 1.0.0 series.

2.2.4 Downloading Source Code

Fired up the terminal because you need to install the qt4 development libraries, tools, compiler and git.

$ sudo apt-get install g++ gcc make git-core libqt4-dev qt4-qmake libqt4-help qt4-dev-tools libboost-all-dev libmuparser-dev libfreetype6-dev

$ sudo apt-get build-dep librecad

Clone the git repository of LibreCAD in Desktop (You can use any Directory)

$ git clone https://github.com/gaganjyot/LibreCAD_3.git

Now you can run qmake (or qmake-qt4) to create a makefile and run make to compile LibreCAD. Make sure that you are in the folder (Librecad).

$ cd LibreCAD

$ qmake-qt4 librecad.pro

$ make

librecad.pro is a project file. qmake creates a makefile. Make command will compile the project. Compiling LibreCAD might take a while, depending on the speed of your computer, but just let it run until it finishes.

To finally run LibreCAD, execute the following commands:

$ cd unix

$ ./librecad

CHAPTER 3

PROJECT WORK

3.1 Text Editor using Qt

This text editor is made using Qt. Qt is the C++ framework. It has most used features that a text editor have. As my 6 weeks training project, I worked in Qt creater and made a text editor. Qt Creator is a complete IDE for creating applications with Qt Quick and the Qt application framework.

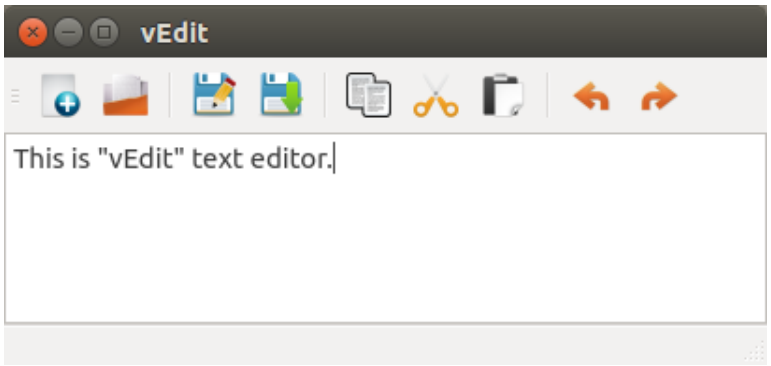

Figure 3.1: vEdit- The Text Editor

It has the following features:

- New File
- Open File
- Save File
- Save as File
- Copy
- Paste

- Cut
- Undo
- Redo

In this text editor we can make new text file and edit text. We can copy, cut and paste text easily. We can save and our text document in any location and open text document from any location.

3.1.1 Installation Guide

To install vEdit, you need to clone it from github.

- Go to terminal and type

  $ git clone http://www.github.com/mandeeps708/vEdit.git

- Now go to the directory vEdit by using:

  $ cd vEdit

- Now running qmake command:

  $ qmake vEdit.pro

- Now MakeFile will be generated. After that run make:

  $ make

- An executable file will be generated. Execute it using:

  $ ./vEdit

3.2 Adding Features to LibreCAD v3

During my 6 weeks training, I worked upon open source, learned a lot from it and in return contributed to it. LibreCAD v3 is written in C++ using Qt framework. It is a Qt application. With this project, I aimed at understanding C++ more clearly and deeply.

I added these features in LibreCAD v3's User Interface:

1. Open
2. Save
3. Save As

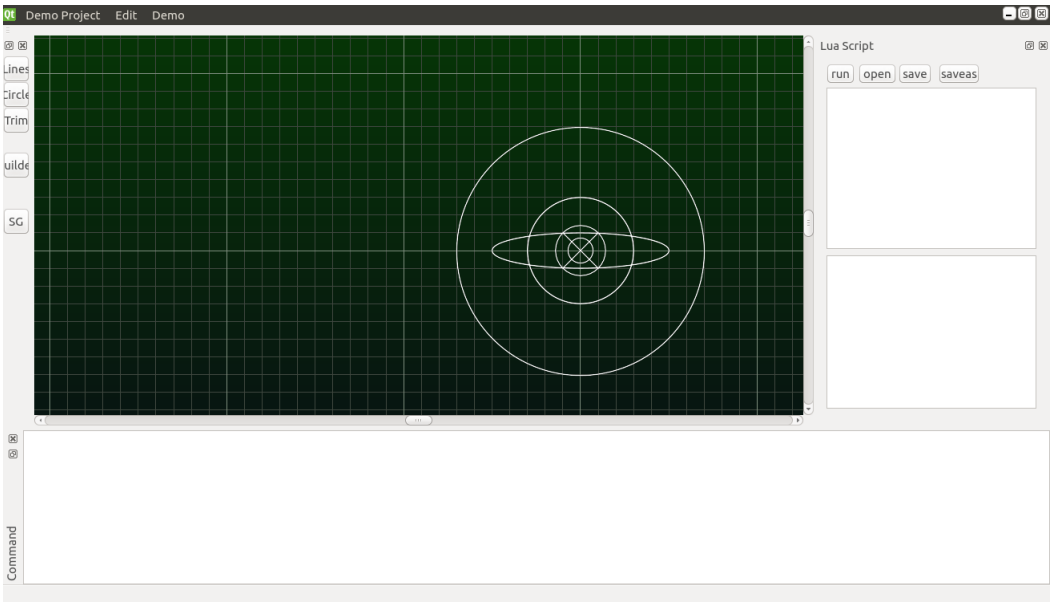

Figure 3.2: LibreCAD v3 Load/Save Feature

3.2.1 Open

In this, I added Open feature to LibreCAD v3. This adds a button to the User Interface. When it is clicked, it prompts for a file to be opened. Then choose existing file and click "open". The file will be opened.

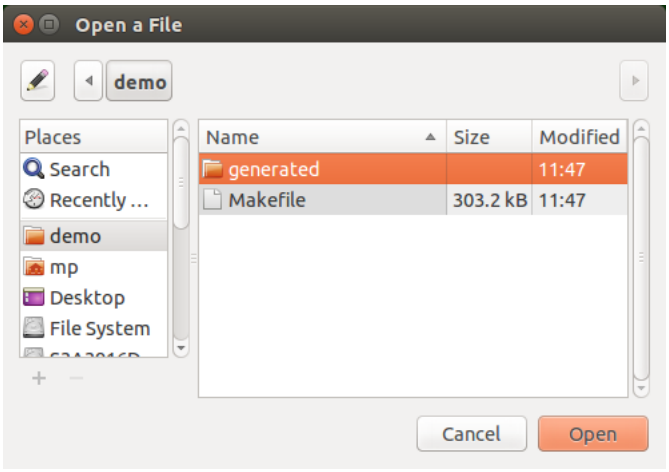

Figure 3.3: Open a File Prompt Box

3.2.2 Save As

[!ht] In this feature, when user clicks on "Save As" button, it prompts for a file location on which the file is to be saved.

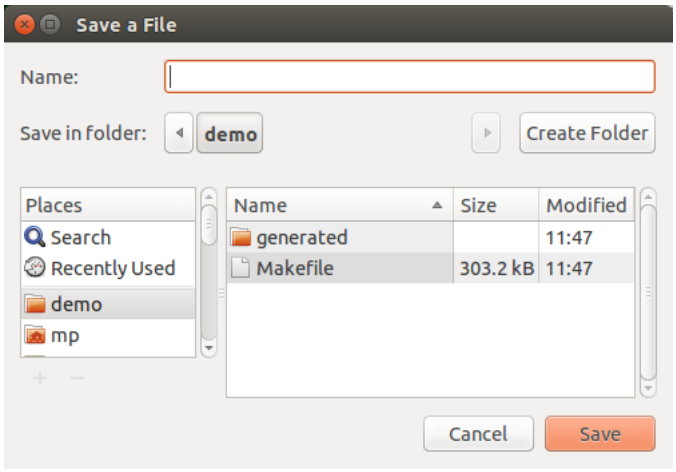

Figure 3.4: Save As Prompt Box

3.2.3 Save

In this feature, when user clicks on save button, it saves the script. It saves the changes to the existing file.

3.3 Design implementation using Lua scripting language

After working with libreCAD 3, I experimented with it by making some designs usign Lua scripting language. In this project, I made several lua scripts and run it in LibreCAD to make various design. I made design of water tank and varoius logos and some other impressive 2D design in LibreCAD using LUA scripting language. In LibreCAD v3, there is a dockable window for creating lua scipts:

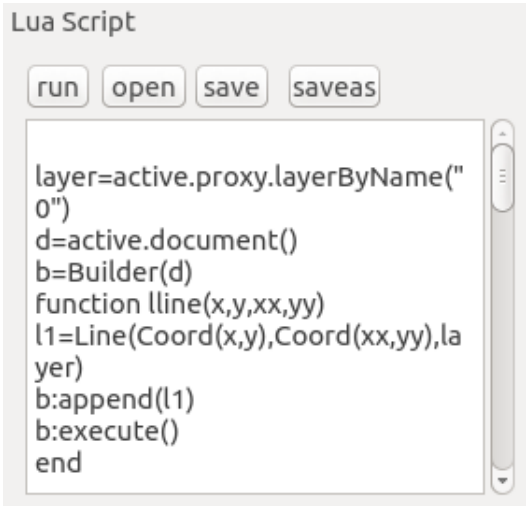

Figure 3.5: Lua Dockable window

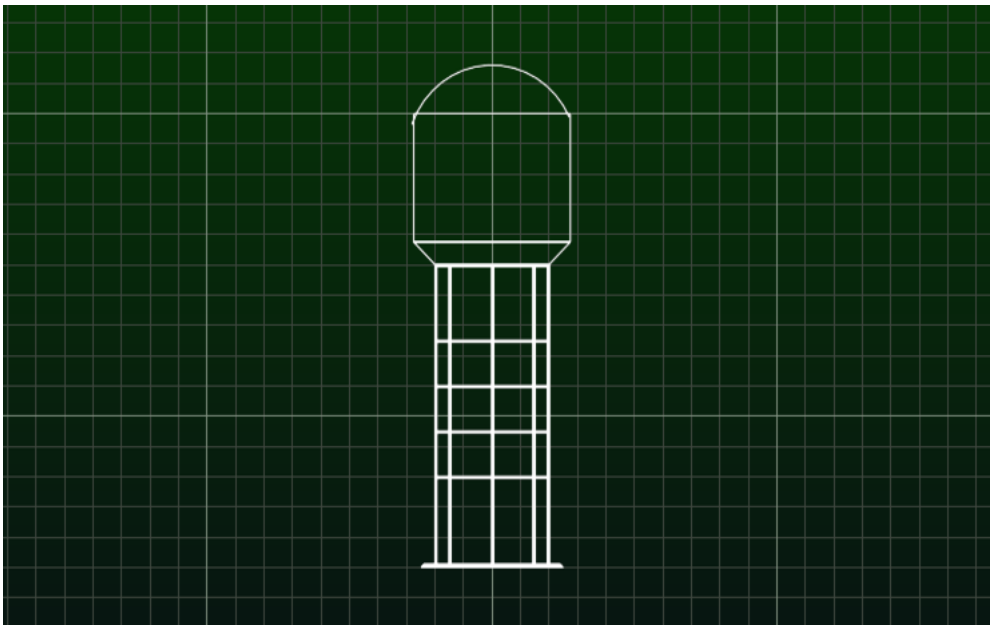

Figure 3.6: Water Tank Model

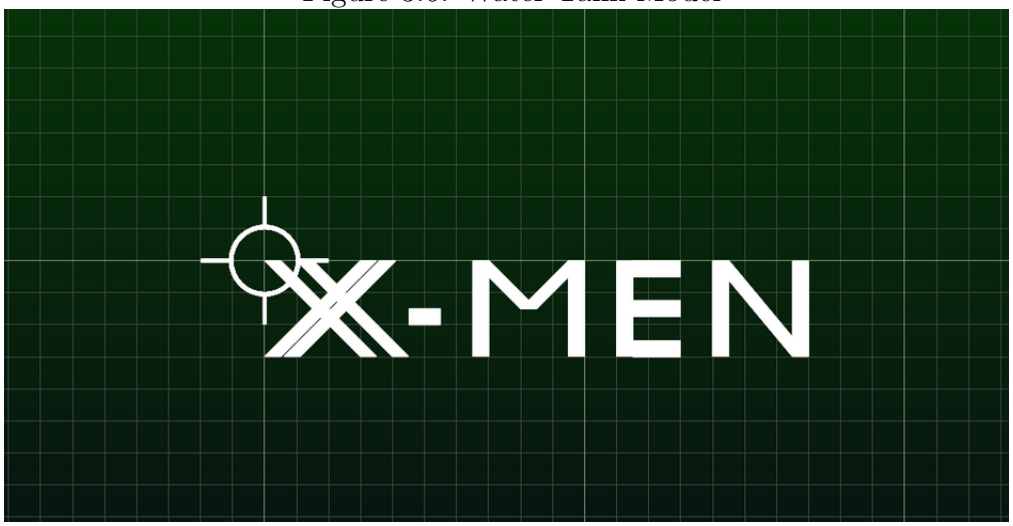

Figure 3.7: X-Men Logo

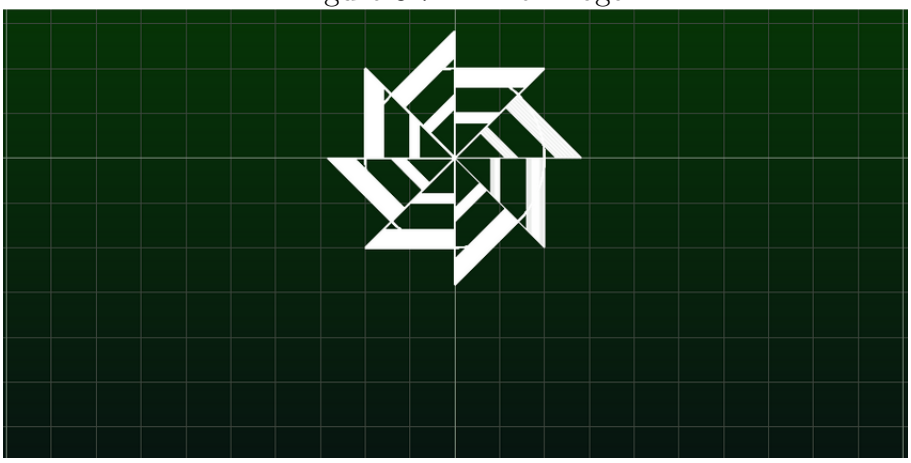

Figure 3.8: Sharp Design 2

3.4 Hatching

This project also involves hatching implementation. To accomplish this, C++ and a graphics library named "Cairo" is used. Basically, Hatching is a pattern of lines and dots that fills an enclosed area. Hatching can be used simply for color. The colors can represent materials or uses in drawings and renderings. There aren't any conventions for use of color in architecture. However, many other disciplines, such as urban planning and cartography, prescribe uses for colors in drawings and maps. During the training, I also worked on Hatching. Basically, I have used C++ and a graphics library called Cairo. In this project, I've implemented two types of hatching:

1. Solid fill
2. Pattern fill

3.4.1 Solid fill

- For Solid Fill, I made a C++ program named "solid.cpp" with help of Cairo graphics library. When it is compiled using:
  $ g++ solid.cpp -o output -lcairo
- Then the executable will be created named "output". Execute it using:
  $ ./output
- Then a file will be created named "hatch.svg" with the given color. We can simply open it in terminal using command:
  $ eog hatch.svg

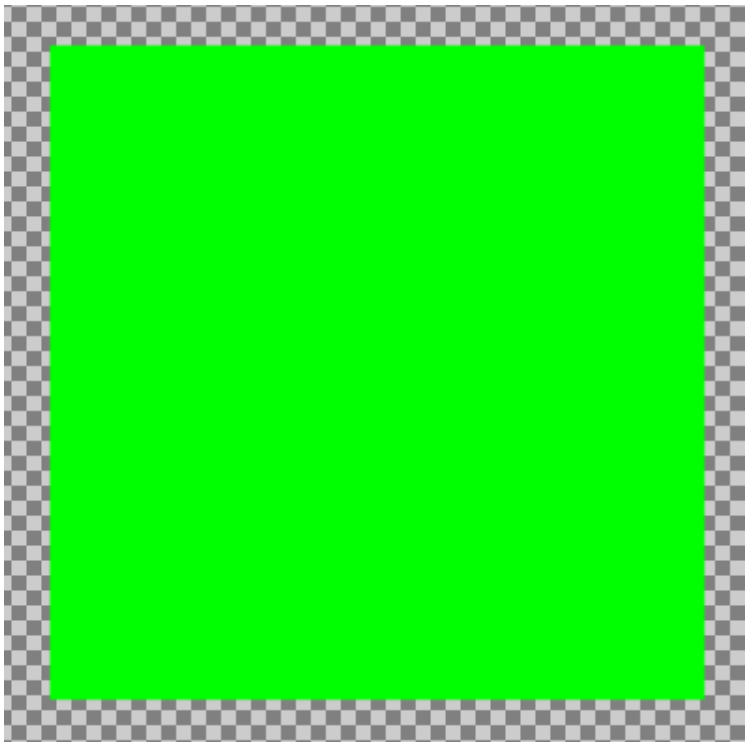

Figure 3.9: Solid Fill: hatch.svg

3.4.2 Pattern Fill

- For Pattern Filling, I made another program named "pattern.cpp". When it is compiled using:
  $ g++ pattern.cpp -o pexec -lcairo

- Then an executable file will be created named "pexec" in the current directory. Execute it using command:
  $ ./pexec

- Then a file will be created named "hello.png" which contains the hatched object. Open this file using:
  $ eog hello.png

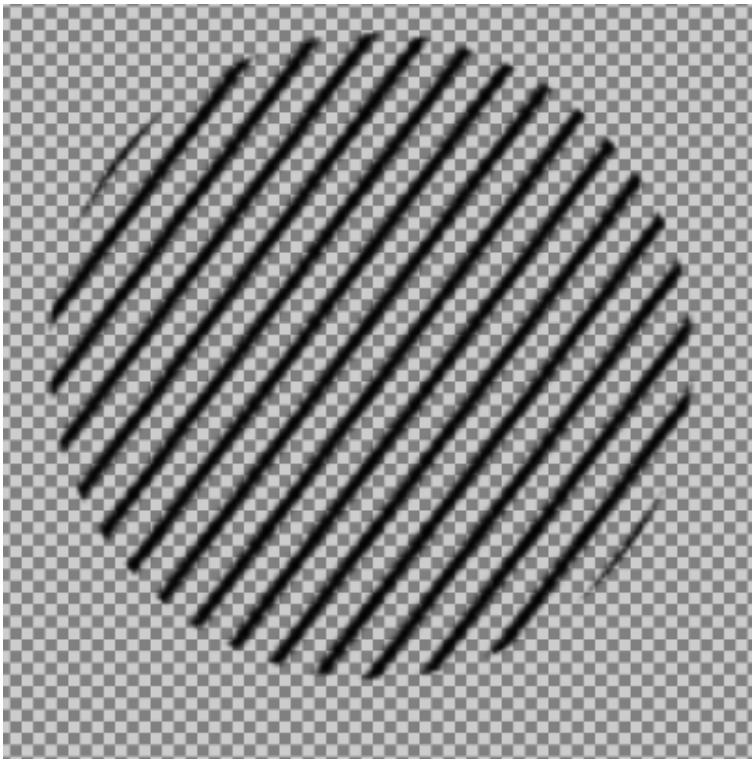

Figure 3.10: Pattern Fill: hello.png

CHAPTER 4

TECHNOLOGIES USED

4.1 Introduction to Ruby On Rails

Before we ride on Rails, let us recapitulate a few points of Ruby, which is the base of Rails.

4.1.1 What is Ruby?

Ruby is a powerful, flexible programming language you can use in web/Internet development, to process text, to create games, and as part of the popular Ruby on Rails web framework. Ruby is:

- **High-level**,
  meaning reading and writing Ruby is really easyit looks a lot like regular English!
- **Interpreted**,
  meaning you don't need a compiler to write and run Ruby. You can write it here at Codecademy or even on your own computer (many are shipped with the Ruby interpreter built inwe'll get to the interpreter later in this lesson).
- **Object-oriented**,
  meaning it allows users to manipulate data structures called objects in order to build and execute programs. We'll learn more about objects later, but for now, all you need to know is everything in Ruby is an object.
- **Easy to use.**
  Ruby was designed by Yukihiro Matsumoto (often just called "Matz") in 1995. Matz set out to design a language that emphasized human needs over those of the computer, which is why Ruby is so easy to pick up

4.1.2 What is Rails?

Rails is a web application development framework written in the Ruby language. It is designed to make programming web applications easier by making assumptions about what every developer needs to get started. It allows you to write less code while accomplishing more than many other

languages and frameworks. Experienced Rails developers also report that it makes web application development more fun.

Rails is opinionated software. It makes the assumption that there is the "best" way to do things, and it's designed to encourage that way - and in some cases to discourage alternatives. If you learn "The Rails Way" you'll probably discover a tremendous increase in productivity. If you persist in bringing old habits from other languages to your Rails development, and trying to use patterns you learned elsewhere, you may have a less happy experience.

The Rails philosophy includes two major guiding principles:

- **Don't Repeat Yourself:**
  DRY is a principle of software development which states that "Every piece of knowledge must have a single, unambiguous, authoritative representation within a system." By not writing the same information over and over again, our code is more maintainable, more extensible, and less buggy.

- **Convention Over Configuration:**
  Rails has opinions about the best way to do many things in a web application, and defaults to this set of conventions, rather than require that you specify every minutiae through endless configuration files.

4.1.3 Installation of Ruby on Rails 4

Start with your platforms native package management system, be it apt-get , dpkg , portage , rpm , rug , synaptic , up2date , or yum . The first step is to install the necessary dependencies. The following instruc- tions are for Ubuntu 13.04 (Raring Ringtail); if youre on a different operating system, you may need to adjust both the command and the package names.

$sudoapt - getinstallapache2curlgitlibmysqlclient - devmysql - servernodejs$

While the Rails core team recommends Ruby 2.0 for use with Rails 4.0, if you want to use a system-installed version of Ruby, you can use Ruby 1.9.3. This will get you up and running quickly. Starting with Ubuntu 12.04, you can install Ruby 1.9.3 and Rails 4.0 with the following commands:
$sudoapt - getinstallruby1.9.3$
sudo gem install rails –version 4.0.0 –no-ri –no-rdoc

Many people prefer instead to have a separate installation of Ruby on their machine dedicated to support their application, and therefore they choose to download and build Ruby. The easiest way weve found to do this is to use RVM. Installing RVM is described on the RVM site. 6 An overview of the steps is included here.
First, install RVM.
$curl - Lhttps : //get.rvm.io|bash - sstable$
rvm requirements –autolibs=enable

Once this is complete, you can proceed to install the Ruby interpreter.

rvm install 2.0.0

This step will take a while as it downloads, configures, and compiles the necessary executables. Once it completes, use that environment, and install rails.

rvm use 2.0.0
gem install rails –version 4.0.0 –no-ri –no-rdoc
With the exception of the rvm use statement, each of the previous instructions needs to be done only once. The rvm use statement needs to be repeated each time you open a shell window. The use keyword is optional, so you can abbreviate this to rvm 2.0.0 . You can also choose to make it the default Ruby interpreter for new Terminal sessions with the following command:
rvm –default 2.0.0
You can verify successful installation using the following command:
rails -v
If you have trouble, try the suggestions listed under the Troubleshooting Your Install heading on the RVM site. 8

4.1.4 Creating a new Rails application

When you install the Rails framework, you also get a new command-line tool, rails , that is used to construct each new Rails application you write. Why do we need a tool to do this? Why cant we just hack away in our favorite editor and create the source for our application from scratch? Well, we could just hack. After all, a Rails application is just Ruby source code. But Rails also does a lot of magic behind the curtain to get our applications to work with a minimum of explicit configuration. To get this magic to work, Rails needs to find all the various components of your application. As well see later (in Section 18.1, Where Things Go, on page 261), this means we need to create a specific directory structure, slotting the code we write into the appropriate places. The rails command simply creates this directory structure for us and populates it with some standard Rails code. To create your first Rails application, pop open a shell window, and navigate to a place in your filesystem where you want to create your applications directory structure. In our example, well be creating our projects in a direc- tory called work . In that directory, use the rails command to create an application called demo . Be slightly careful hereif you have an existing directory called demo , you will be asked whether you want to overwrite any existing files. (Note: if you want to specify which Rails version to use, as described in Section 1.4, Choosing a Rails Version, on page 8, now would be the time to do so.)

cd work
rails new demo
create
create README.rdoc
create Rakefile
create config.ru
:
:
:
create vendor/assets/stylesheets
create vendor/assets/stylesheets/.keep
run bundle install
Fetching gem metadata from https://rubygems.org/.....

:
:
:
Your bundle is complete!
Use ‘bundle show [gemname]‘ to see where a bundled gem is installed.
work¿ The command has created a directory named demo . Pop down into that directory, and list its contents (using ls on a Unix box or using dir under Windows). You should see a bunch of files and subdirectories.
work¿
demo¿
app/
bin/
cd demo
ls -p
config/
config.ru
db/
Gemfile
Gemfile.lock
lib/
log/
public/
Rakefile
README.rdoc
test/
tmp/
vendor/

All these directories (and the files they contain) can be intimidating to start with, but we can ignore most of them for now. In this chapter, well use only one of them directly: the app directory, where well write our application. Examine your installation using the following command:
demo¿ rake about If you get a Rails version other than 4.0.0 This command will also detect common installation errors. For example, if it cant find a JavaScript runtime, it will provide you with a link to available runtimes. If you see a bunch of messages concerning already initialized constants or a possible conflict with an extension, consider deleting the demo directory, cre- ating a separate RVM gemset, 1 and starting over. If that doesnt work, use bundle exec to run rake commands. Once you get rake about working, you have everything you need to start a stand- alone web server that can run our newly created Rails application. So, without further ado, lets start our demo application.

demo rails server
= Booting WEBrick
= Rails 4.0.0 application starting in development on http://0.0.0.0:3000
= Run ‘rails server -h‘ for more startup options
= Ctrl-C to shutdown server

Which web server is run depends on what servers you have installed. WEBrick is a pure-Ruby

web server that is distributed with Ruby itself and therefore is guaranteed to be available. However, if another web server is installed on your system (and Rails can find it), the rails server command may use it in preference to WEBrick. You can force Rails to use WEBrick by providing an option to the rails command.

rails server webrick

As the last line of the startup tracing indicates, we just started a web server on port 3000. The 0.0.0.0 part of the address means that WEBrick will accept connections on all interfaces. On Daves OS X system, that means both local interfaces (127.0.0.1 and ::1) and his LAN connection. We can access the application by pointing a browser at the URL http://localhost:3000. The result is shown in Figure 1, Newly created Rails application, on page 18. If you look at the window where you started the server, youll see tracing showing you started the application. Were going to leave the server running in this console window. Later, as we write application code and run it via our browser, well be able to use this console window to trace the incoming requests. When the time comes to shut down your application, you can press Ctrl-C in this window to stop WEBrick. (Dont do that yetwell be using this particular application in a minute.) At this point, we have a new application running, but it has none of our code in it. Lets rectify this situation.

4.2 Introduction to MVC

Back in 1979, Trygve Reenskaug came up with a new architecture for developing interactive applications. In his design, applications were broken into three types of components: models, views, and controllers.
The **model** is responsible for maintaining the state of the application. Some- times this state is transient, lasting for just a couple of interactions with the user. Sometimes the state is permanent and will be stored outside the appli- cation, often in a database. A model is more than just data; it enforces all the business rules that apply to that data.

The **view** is responsible for generating a user interface, normally based on data in the model. For example, an online store will have a list of products to be displayed on a catalog screen. This list will be accessible via the model, but it will be a view that formats the list for the end user. Although the view may present the user with various ways of inputting data, the view itself never handles incoming data. The views work is done once the data is dis- played. There may well be many views that access the same model data, often for different purposes. In the online store, therell be a view that displays product information on a catalog page and another set of views used by administrators to add and edit products. Controllers orchestrate the application. Controllers receive events from the outside world (normally user input), interact with the model, and display an appropriate view to the user.

This triumviratethe model, view, and controllertogether form an architecture known as MVC. To learn how the three concepts fit together, see the following figure The MVC architecture was originally intended for conventional GUI applications, where developers found the separation of concerns led to far less coupling, which in turn made the code easier to write and maintain. Each concept or action was expressed in just one well-known place. Using MVC was like constructing a skyscraper with the girders already in placeit was a lot easier to hang the rest of the pieces with a structure already there. During the development of our application, we will be making heavy use of Rails ability to generate scaffolding for our application. Ruby on Rails is an MVC framework, too. Rails enforces a structure for your applicationyou develop models, views, and controllers as separate chunks of functionality, and it knits them together as your program executes. One of the joys of Rails is that this knitting process is based on the use of intelligent defaults so that you typically dont need to write any external configuration metadata to make it all work. This is an example of the Rails philosophy of favoring convention over configuration. In a Rails application, an incoming request is first sent to a router, which works out where in the application the request should be sent and how the request itself should be parsed. Ultimately, this phase identifies a particular method (called an action in Rails parlance) somewhere in the controller code. The action might look at data in the request, it might interact with the model, and it might cause other actions to be invoked. Eventually the action prepares information for the view, which renders something to the user. Rails handles an incoming request as shown in the following figure. In this example, the application has previously displayed a product catalog page, and the user has just clicked the Add to Cart button next to one of the products.

By convention, POST methods are associated with create() actions. As a result of all this analysis, the router knows it has to invoke the create() method in the controller class LineItemsController The create() method handles user requests. In this case, it finds the current users shopping cart (which is an object managed by the model). It also asks the model to find the

information for product 2. It then tells the shopping cart to add that product to itself. (See how the model is being used to keep track of all the business data? The controller tells it what to do, and the model knows how to do it.) Now that the cart includes the new product, we can show it to the user. The controller invokes the view code, but before it does, it arranges things so that the view has access to the cart object from the model. In Rails, this invocation is often implicit; again, conventions help link a particular view with a given action.
Thats all there is to an MVC web application. By following a set of conventions and partitioning your functionality appropriately, youll discover that your code becomes easier to work with and your application becomes easier to extend and maintain. That seems like a good trade. If MVC is simply a question of partitioning your code a particular way, you might be wondering why you need a framework such as Ruby on Rails. The answer is straightforward: Rails handles all of the low-level housekeeping for youall those messy details that take so long to handle by yourselfand lets you concentrate on your applications core functionality. Lets see how.

4.3 Sass (Syntactically Awesome StyleSheets)

Sass is an extension of CSS that adds power and elegance to the basic language. It allows you to use variables, nested rules, mixins, inline imports, and more, all with a fully CSS-compatible syntax. Sass helps keep large stylesheets well-organized, and get small stylesheets up and running quickly, particularly with the help of the Compass style library.

Features

- Fully CSS-compatible
- Language extensions such as variables, nesting, and mixins
- Many useful functions for manipulating colors and other values
- Advanced features like control directives for libraries
- Well-formatted, customizable output

Syntax

There are two syntaxes available for Sass. The first, known as **SCSS (Sassy CSS)** and used throughout this reference, is an extension of the syntax of CSS. This means that every valid CSS stylesheet is a valid SCSS file with the same meaning. In addition, SCSS understands most CSS hacks and vendor-specific syntax, such as IEs old filter syntax. This syntax is enhanced with the Sass features described below. Files using this syntax have the .scss extension.

The second and older syntax, known as the indented syntax (or sometimes just Sass), provides a more concise way of writing CSS. It uses indentation rather than brackets to indicate nesting of selectors, and newlines rather than semicolons to separate properties. Some people find this to be easier to read and quicker to write than SCSS. The indented syntax has all the same features, although some of them have slightly different syntax; this is described in the indented syntax reference. Files using this syntax have the .sass extension.

Either syntax can import files written in the other. Files can be automatically converted from one syntax to the other using the sass-convert command line tool:

Convert Sass to SCSS

sass-convert style.sass style.scss

Convert SCSS to Sass

sass-convert style.scss style.sass

Note that this command does not generate CSS files. For that, use the sass command described elsewhere.

Using Sass Sass can be used in three ways: as a command-line tool, as a standalone Ruby module, and as a plugin for any Rack-enabled framework, including Ruby on Rails and Merb. The first step for all of these is to install the Sass gem:

gem install sass

If youre using Windows, you may need to install Ruby first.

To run Sass from the command line, just use

sass input.scss output.css

You can also tell Sass to watch the file and update the CSS every time the Sass file changes:
sass –watch input.scss:output.css

If you have a directory with many Sass files, you can also tell Sass to watch the entire directory:
sass –watch app/sass:public/stylesheets
Use sass –help for full documentation.
Using Sass in Ruby code is very simple. After installing the Sass gem, you can use it by running require "sass" and using Sass::Engine like so:
Rack/Rails/Merb Plugin

To enable Sass in Rails versions before Rails 3, add the following line to environment.rb:
config.gem "sass"

For Rails 3, instead add the following line to the Gemfile:
gem "sass"

To enable Sass in Merb, add the following line to config/dependencies.rb:
dependency "merb-haml"

To enable Sass in a Rack application, add the following lines to config.ru.
require 'sass/plugin/rack' use Sass::Plugin::Rack

4.4 Introduction to LaTeX

LaTeX, I had never heard about this term before doing this project, but when I came to know about its features, found it excellent. LaTeX (pronounced /letk/, /letx/, /ltx/, or /ltk/) is a document markup language and document preparation system for the TeX typesetting program. Within the typesetting system, its name is styled as LaTeX.

Figure 4.1: Donald Knuth, Inventor Of TeX typesetting system

Within the typesetting system, its name is styled as LaTeX. The term LaTeX refers only to the language in which documents are written, not to the editor used to write those documents. In order to create a document in LaTeX, a .tex file must be created using some form of text editor. While most text editors can be used to create a LaTeX document, a number of editors have been created specifically for working with LaTeX.

LaTeX is most widely used by mathematicians, scientists, engineers, philosophers, linguists, economists and other scholars in academia. As a primary or intermediate format, e.g., translating DocBook and other XML-based formats to PDF, LaTeX is used because of the high quality of typesetting achievable by TeX. The typesetting system offers programmable desktop publishing features and extensive facilities for automating most aspects of typesetting and desktop publishing, including numbering and cross-referencing, tables and figures, page layout and bibliographies.

LaTeX is intended to provide a high-level language that accesses the power of TeX. LaTeX essentially comprises a collection of TeX macros and a program to process LaTeXdocuments. Because the TeX formatting commands are very low-level, it is usually much simpler for end-users to use LaTeX.

4.4.1 Typesetting

LaTeX is based on the idea that authors should be able to focus on the content of what they are writing without being distracted by its visual presentation. In preparing a LaTeX document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the LaTeX system worry about the presentation of these structures. It therefore en-

courages the separation of layout from content while still allowing manual typesetting adjustments where needed.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{\LaTeX}
\date{}
\begin{document}
  \maketitle
  \LaTeX{} is a document preparation system
  for the \TeX{} typesetting program.
   \par
   $E=mc^2$
\end{document}
```

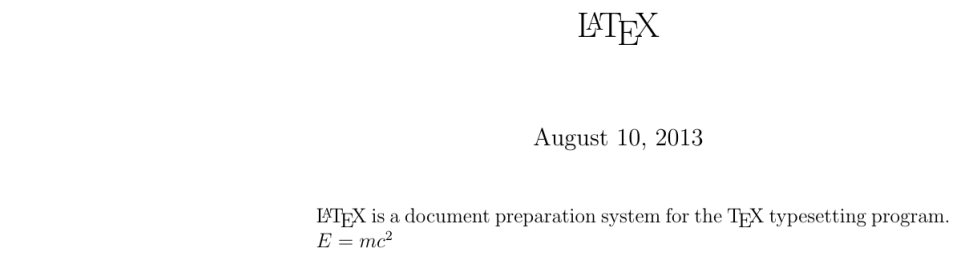

Figure 4.2: LaTeX output of above program.

4.4.2 Installing LaTeX on System

Installation of LaTeX on personal system is quite easy. As i have used LaTeX on Ubuntu 13.04 so i am discussing the installation steps for Ubuntu 13.04 here:

- Go to terminal and type

  $ sudo apt-get install texlive-full

- Your Latex will be installed on your system and you can check for manual page by typing.

  $ man latex
  in terminal which gives manual for latex command.

- To do very next step now one should stick this to mind that the document which one is going to produce is written in any type of editor whether it may be your most common usable editor Gedit or you can use vim by installing first vim into your system using command.

  $ sudo apt-get install vim

- After you have written your document it is to be embedded with some set of commands that Latex uses so as to give a structure to your document. Note that whenever you wish your document to be looked into some other style just change these set of commands.

- When you have done all these things save your piece of code with .tex format say test.tex. Go to terminal and type

  latex path of the file test.tex Or pdflatex path of the file test.tex
  eg: pdflatex test.tex
  for producing pdf file simultaneously.
  After compiling it type command

  $ evince filename.pdf
  eg: evince test.pdf
  To see output pdf file.

4.4.3 Graphical Editors for LaTeX

LaTeX is not restricted to command line only there are so many graphical based editors available to be used. These GUi based editors provide an easy interface to user so as to do typesetting in an efficient manner. Some of them are listed below:

- Texmaker

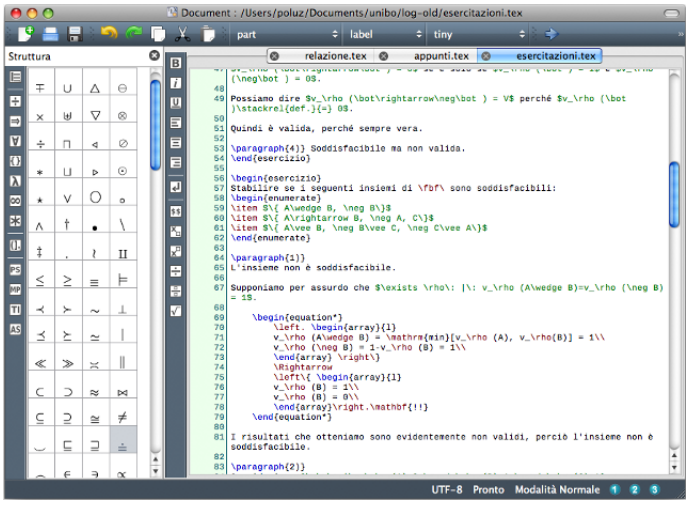

Figure 4.3: Texmaker, A Graphical LaTeX Editor

- LEd

And many more but the preferred method to produce LaTeX document is through console mode only.

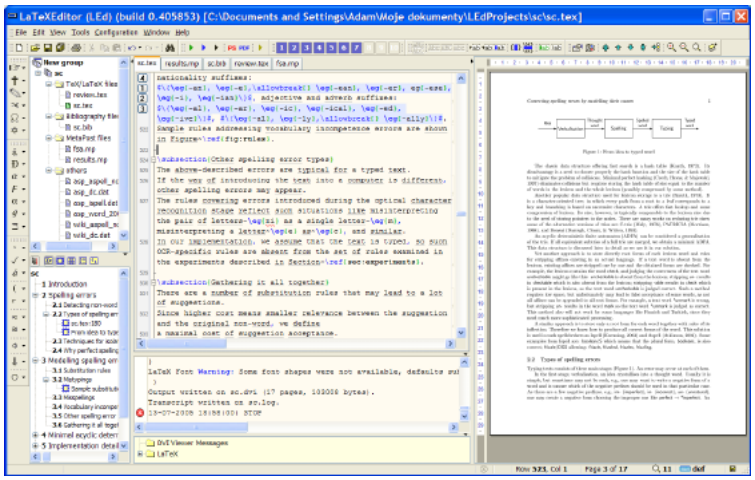

Figure 4.4: LEd, A Graphical LaTeX Editor

4.4.4 Pdfscreen LaTeX

There are some packages that can help to have unified document using LaTeX. Example of such a package is pdfscreen that let the user view its document in two forms-print and screen. Print for hard copy and screen for viewing your document on screen. Download this package from www.ctan.org/tex-archive/macros/latex/contrib/pdfscreen/.
Then install it using above mention method.

Just changing print to screen gives an entirely different view. But for working of pdfscreen another package required are comment and fancybox.

The fancybox package provides several different styles of boxes for framing and rotating content in your document. Fancybox provides commands that produce square-cornered boxes with single or double lines, boxes with shadows, and round-cornered boxes with normal or bold lines. You can box mathematics, floats, center, flushleft, and flushright, lists, and pages.

Whereas comments package selectively include/excludes portions of text. The comment package allows you to declare areas of a document to be included or excluded. One need to make these declarations in the preamble of your file. The package uses a method for exclusion that is pretty robust, and can cope with ill-formed bunches of text.

So these extra packages needed to be installed on system for the proper working of pdfscreen package.

4.4.5 Web based graphic generation using LaTeX

LaTeX is also useful when there is need of generating the graphics from browser. For example to draw a circle by just entering its radius in html input box. So this kind A of project can be conveniently handled using LaTeX. Basic idea behind this generation process is that when user clicks on submit button after entering radius a script will run that enter the radius in already made .tex file and recompiles it on server and makes its pdf and postscript file. After that user can view those files by clicking on link provided to view the files. See some screen shots of such a graphic

generation project made by Dr. H.S. Rai:

So here in the above input page which is also the index page user can enter input for length of rectangle, breadth of rectangle and for radius of circle after that user can submit the values. After the values get submitted a script get runs by php code at server side. This script first enters the dimensions of rectangle and circle that were selected by user in to an already existing .tex file and replace with the older dimensions there. After that script recompiles the the tex file and make it available for user.

In above figure it gets clear that .tex file has been compiled and pdf and postscript files are available to user and user can download the graphics so produced. Hence graphics can be generated in LaTeX through web interface.

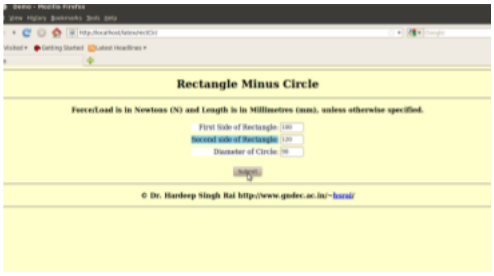

Figure 4.5: Web based graphic generation using LaTeX(input page)

4.5 Introduction to Lua scripting language

Figure 4.6: Lua Logo

Lua is a powerful and fast programming language that is easy to learn and use and to embed into your application. Lua is designed to be a lightweight embeddable scripting language and is used for all sorts of applications from games to web applications and image processing.

Lua combines simple procedural syntax with powerful data description constructs based on associative arrays and extensible semantics. Lua is dynamically typed, runs by interpreting bytecode

for a register-based virtual machine, and has automatic memory management with incremental garbage collection, making it ideal for configuration, scripting, and rapid prototyping.

4.5.1 Installation of Lua

Lua is implemented in pure ANSI C and compiles unmodified in all platforms that have an ANSI C compiler. Lua also compiles cleanly as C++. The current version is lua-5.2.3. Just download the source from the lua.org/download and then follow the steps for building it

$ curl -R -O http://www.lua.org/ftp/lua-5.2.3.tar.gz
$ tar zxf lua-5.2.3.tar.gz
$cd lua-5.2.3
$make

4.5.2 Why choose Lua?

- **Lua is a proven, robust language**
  Lua has been used in many industrial applications (e.g., Adobe's Photoshop Lightroom), with an emphasis on embedded systems (e.g., the Ginga middleware for digital TV in Brazil) and games (e.g., World of Warcraft and Angry Birds). Lua is currently the leading scripting language in games. Lua has a solid reference manual and there are several books about it. Several versions of Lua have been released and used in real applications since its creation in 1993. Lua featured in HOPL III, the Third ACM SIGPLAN History of Programming Languages Conference, in June 2007.

- **Lua is fast**
  Lua has a deserved reputation for performance. To claim to be "as fast as Lua" is an aspiration of other scripting languages. Several benchmarks show Lua as the fastest language in the realm of interpreted scripting languages. Lua is fast not only in fine-tuned benchmark programs, but in real life too. Substantial fractions of large applications have been written in Lua.

- **Lua is portable**
  Lua is distributed in a small package and builds out-of-the-box in all platforms that have a standard C compiler. Lua runs on all flavors of Unix and Windows, on mobile devices (running Android, iOS, BREW, Symbian, Windows Phone), on embedded microprocessors (such as ARM and Rabbit, for applications like Lego MindStorms), on IBM mainframes, etc.

- **Lua is embeddable**
  Lua is a fast language engine with small footprint that you can embed easily into your application. Lua has a simple and well documented API that allows strong integration with code written in other languages. It is easy to extend Lua with libraries written in other languages. It is also easy to extend programs written in other languages with Lua. Lua has been used to extend programs written not only in C and C++, but also in Java, C#,

Smalltalk, Fortran, Ada, Erlang, and even in other scripting languages, such as Perl and Ruby.

- **Lua is powerful (but simple)**
  A fundamental concept in the design of Lua is to provide meta-mechanisms for implementing features, instead of providing a host of features directly in the language. For example, although Lua is not a pure object-oriented language, it does provide meta-mechanisms for implementing classes and inheritance. Lua's meta-mechanisms bring an economy of concepts and keep the language small, while allowing the semantics to be extended in unconventional ways.

- **Lua is small**
  Adding Lua to an application does not bloat it. The tarball for Lua 5.2.3, which contains source code and documentation, takes 246K compressed and 960K uncompressed. The source contains around 20000 lines of C. Under Linux, the Lua interpreter built with all standard Lua libraries takes 182K and the Lua library takes 244K.

- **Lua is free**
  Lua is free open-source software, distributed under a very liberal license (the well-known MIT license). It may be used for any purpose, including commercial purposes, at absolutely no cost. Just download it and use it.

4.6 Introduction to Cairo Graphics Library

Cairo is a library for creating 2D vector graphics. It is written in the C programming language. Bindings for other computer languages exist. Python, Perl, C++, C#, Java. Cairo is a multiplatform library, it works on Linux, BSDs, and Mac OS.

Figure 4.7: Cairo Logo

Cairo supports various backends:

- X Window System
- Win32 GDI
- Mac OS X Quartz
- PNG
- PDF
- PostScript
- SVG

This means that we can use the library to draw on windows on Linux/BSDs, Windows, Mac OS and we can use the library to create PNG images, PDF files, PostScript files, and SVG files.

4.7 Introduction to Github

GitHub is a Git repository web-based hosting service which offers all of the functionality of Git as well as adding many of its own features. Unlike Git which is strictly a command-line tool, Github provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project.

GitHub offers both paid plans for private repto handle everything from small to very large projects with speed and efficiency. ositories, and free accounts, which are usually used to host open source software projects. As of 2014, Github reports having over 3.4 million users, making it

Figure 4.8: Github Logo

the largest code host in the world.

GitHub has become such a staple amongst the open-source development community that many developers have begun considering it a replacement for a conventional resume and some employers require applications to provide a link to and have an active contributing GitHub account in order to qualify for a job.

The Git feature that really makes it stand apart from nearly every other Source Code Management (SCM) out there is its branching model.

Git allows and encourages you to have multiple local branches that can be entirely independent of each other. The creation, merging, and deletion of those lines of development takes seconds.

This means that you can do things like:

- Frictionless Context Switching.
  Create a branch to try out an idea, commit a few times, switch back to where you branched from, apply a patch, switch back to where you are experimenting, and merge it in.
- Role-Based Codelines.
  Have a branch that always contains only what goes to production, another that you merge work into for testing, and several smaller ones for day to day work.
- Feature Based Workflow.
  Create new branches for each new feature you're working on so you can seamlessly switch back and forth between them, then delete each branch when that feature gets merged into your main line.
- Disposable Experimentation.
  Create a branch to experiment in, realize it's not going to work, and just delete it - abandoning the workwith nobody else ever seeing it (even if you've pushed other branches in the meantime).

Notably, when you push to a remote repository, you do not have to push all of your branches. You can choose to share just one of your branches, a few of them, or all of them. This tends to free people to try new ideas without worrying about having to plan how and when they are going to merge it in or share it with others.

There are ways to accomplish some of this with other systems, but the work involved is much more difficult and error-prone. Git makes this process incredibly easy and it changes the way most developers work when they learn it.

4.7.1 What is Git?

Figure 4.9: Git Logo

Git is a distributed revision control and source code management (SCM) system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows. Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since become the most widely adopted version control system for software development.

As with most other distributed revision control systems, and unlike most clientserver systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. Like the Linux kernel, Git is free and open source software distributed under the terms of the GNU General Public License version 2 to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

4.7.2 Installation of Git

Installation of git is a very easy process. The current git version is: 2.0.4. Type the commands in the terminal:

$ sudo apt-get update

$ sudo apt-get install git

This will install the git on your pc or laptop.

4.7.3 Various Git Commands

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. The commonly used Git command line instructions are:-

4.7.3.1 Create Repositories

Start a new repository or obtain from an exiting URL

$ git init [**project-name**]
Creates a new local repository with the specified name

$ git clone [**url**]
Downloads a project and its entire version history

4.7.3.2 Make Changes

Review edits and craft a commit transaction

$ git status
Lists all new or modified files to be committed

$ git diff
Shows file differences not yet staged

$ git add [**file**]
Snapshots the file in preparation for versioning

$ git reset [**file**]
Unstages the file, but preserve its contents

$ git commit -m "[**descriptive message** "]
Records file snapshots permanently in version history

4.7.3.3 Group Changes

Name a series of commits and combine completed efforts

$ git branch
Lists all local branches in the current repository

$ git branch [**branch-name**]
Creates a new branch

$ git checkout [**branch-name**]
Switches to the specified branch and updates the working directory

$ git merge [**branch**]
Combines the specified branchs history into the current branch

$ git branch -d [**branch-name**]
Deletes the specified branch

4.7.3.4 Save Fragments

Shelve and restore incomplete changes

$ git stash
Temporarily stores all modified tracked files

$ git stash pop
Restores the most recently stashed files

$ git stash list
Lists all stashed changesets

$ git stash drop
Discards the most recently stashed changeset

4.7.3.5 Synchronize Changes

Register a repository bookmark and exchange version history

$ git fetch [**bookmark**]
Downloads all history from the repository bookmark

$ git merge [**bookmark** /[branch]]
Combines bookmarks branch into current local branch

$ git push [**alias** [branch]]
Uploads all local branch commits to GitHub

$ git pull
Downloads bookmark history and incorporates changes

CHAPTER 5

PROJECT LEGACY

5.1 Technical and Managerial Lesson Learnt

I learned a lot by doing this project . During this period I got to learn a vast number of technologies. These are listed below :

- **Operating system**: Ubuntu
- **Languages used**: C++, Lua
- **Framework**: Qt
- **Typesetting**: LaTex
- **Other Learnings**: Internet Relay Chat(IRC), Wordpress

So during this project I learned all the above things. Above all I got to know how Softwares are developed from the scratch. Planning, designing, developing code, working in a team, testing etc. These are all very precious things I got to learn during this period.

5.1.1 Ubuntu: An open source OS

During my training, I also got familiar with a great and open source Operating System, Ubuntu. Firstly, it was quite difficult for a regular MS Windows user to port to Ubuntu. I did all of my project work using this vast operating system. Ubuntu (/ubuntu/ oo-BOON-too) is a Debian-based Linux operating system, with Unity as its default desktop environment. It is based on free software and named after the Southern African philosophy of ubuntu (literally, "human-ness"), which often is translated as "humanity towards others" or "the belief in a universal bond of sharing that connects all humanity".

Ubuntu's goal is to be secure "out-of-the box". By default user's programs run with low privileges and cannot corrupt the operating system or other user's files. For increased security, the sudo tool is used to assign temporary privileges for performing administrative tasks, which allows the root account to remain locked and helps prevent inexperienced users from inadvertently making catastrophic system changes or opening security holes.

5.1.2 Internet Relay Chat

Internet Relay Chat (IRC) is an application layer protocol that facilitates transfer of messages in the form of text. The chat process works on a client/server model of networking. IRC clients are computer programs that a user can install on their system. These clients are able to communicate with chat servers to transfer messages to other clients. It is mainly designed for group communication in discussion forums, called channels, but also allows one-to-one communication via private message as well as chat and data transfer, including file sharing. Client software is available for every major operating system that supports Internet access.IRC is an open protocol that uses TCP and, optionally, TLS. An IRC server can connect to other IRC servers to expand the IRC network. Users access IRC networks by connecting a client to a server. There are many client implementations, such as mIRC, HexChat and irssi, and server implementations, e.g. the original IRCd. Most IRC servers do not require users to register an account but a user will have to set a nickname before being connected.

IRC has a line-based structure with the client sending single-line messages to the server,receiving replies to those messages and receiving copies of some messages sent by other clients. In most clients, users can enter commands by prefixing them with a '/'. Depending on the command, these may either be handled entirely by the client, or passed directly to the server, possibly with some modification.

The basic means of communicating to a group of users in an established IRC session is through a channel. Channels on a network can be displayed using the IRC command LIST, which lists all currently available channels that do not have the modes +s or +p set, on that particular network. Users can join a channel using the JOIN command, in most clients available as /join #channel-name. Messages sent to the joined channels are then relayed to all other users.

I used "Xchat IRC" client for IRC during my training to communicate with the Open Source communities whenever I found some error during my work or for some extra knowledge. I communicated with several IRC channels including #librecad, #greatdevelopers etc. XChat is a graphical IRC Client with a GTK+ GUI. It's special features include the multiple server/channel windows, dialog windows, plugin API.

5.1.3 Shell Scripting

Because all my project work was being done in Ubuntu (i.e. a Linux distribution), I came to know the power of Bash(Terminal). We can simply call it a "Shell". It is a great tool to work faster. With terminal, the efficiency of work increases. Ubuntu uses the Bash(Bourne Again Shell) as default shell. During training, I came to know about the new commands daily. Working with CLI(Command Line Interface) is a fun.

5.2 Future Scope

The future of GUI programming in my opinion is Qt programming, which involves the efficiency and low levelness of C++ as its a C++ framework but also has all the qualities that a modern programming language has. It gives you a way to easily develop applications and GUIs which using C++ was earlier not possible. Open source is always growing field I have a vision that

LibreCAD will someday be as famous as other CAD programs such as Autocad. This is the dawn of Open Source Softwares and we still have to grow a lot. I have added some new features and in LibreCAD, the LibreCAD is going to improve day by day. LibreCAD is completely scalable so I can also add as many features as I want. Future scope is bright as I believe that this is just the beginning and there is allot more to learn and implement.

BIBLIOGRAPHY

[1] LibreCAD 3, https://github.com/gaganjyot/LibreCAD_3

[2] LaTeX Beginner's Guide By Stefan Kottwitz [Pact Publishing]

[3] My Blog, http://mandeep7.wordpress.com

[4] My Github Profile, http://github.com/mandeeps708