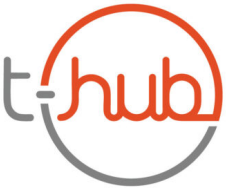


SRISHTI INTERNSHIP REPORT

Software Engineering Research Center(SERC), T-Hub
International Institute of Information Technology, Hyderabad

Submitted by

Rohail Alam
Second Year Student
Indian Institute of Information Technology, Design and Manufacturing,
Kancheepuram

Under the guidance of

Prof. Venkatesh Choppella
Associate Professor
International Institute of Information Technology,
Hyderabad

June 30, 2022

Start Date: 15.05.2022
End Date: 30.06.2022

Acknowledgement

This project was only possible with the esteemed guidance of my supervisor, Prof. Venkatesh Choppella. I'd like to thank him for providing me the wonderful opportunity to work on his research project.

I would also like to thank my mentors, Archit and Prince for their tireless efforts. Their expertise and valuable insight in the field was truly inspirational. I would also like to appreciate my colleagues from various technical backgrounds who have always tried their level best to tackle the tasks assigned to us through brainstorming and collaboration.

I am extremely grateful to the organizers of the SRISHTI Internship program for providing us accommodation to the campus.

I am highly indebted to Prof Priyanka Kokil, IIITDM Kancheepuram for her advice to take up this internship.

This internship has taught me about the importance of brainstorming, teamwork, planning, designing, implementation, testing and debugging, for which I am immensely grateful and it is with a great sense of gratitude that I acknowledge the help of these individuals.

Abstract

Software Engineering Research Center (SERC) has a eminent faculty with vast teaching and research experience in and outside India. SERC has close collaboration with industry providing software services to large organizations, R&D labs of various organizations and other academic institutes (India and abroad). Organizations that SERC is working are diverse, stratified across banking and finance, government, equipment manufacture, ISVs building products for software industry, etc.

Algodynamics : *A new approach to learning algorithms*

Algorithms are the cornerstone of introductory Computer Science courses for engineering students. This is one of the most directly relevant courses for students who join industry after an undergraduate course. However, it is felt that the traditional approach to teaching algorithms in engineering colleges have a few challenges that have still not been addressed.

For example, most algorithm textbooks and classroom are very prescriptive – they describe the algorithm in a particular way using pseudo-code or program and expect the students to understand them as-is, without helping them understand why. Similarly, tinkering and experimenting are critical for learning for engineering students, but these approaches don't get applied for algorithm teaching.

Algodynamics applies system dynamics approach to describe and teach algorithms and aims to address some of these challenges.

For experiencing the pedagogy, visit the Labs or Experiments page.

Contents

1 Introduction **5**
- 1.1 Learning Objectives . . . 5
- 1.2 About Internship . . . 5
- 1.3 Installation and Integration . . . 6

2 Projects **7**
- 2.1 elm-bblsort . . . 7
  - 2.1.1 Assigned Task . . . 7
  - 2.1.2 Implementation . . . 7
  - 2.1.3 Learning Outcomes . . . 8
- 2.2 elm-calci . . . 8
  - 2.2.1 Assigned Task . . . 8
  - 2.2.2 Implementation . . . 8
  - 2.2.3 Learning Outcomes . . . 9
- 2.3 elm-dagre . . . 10
  - 2.3.1 Assigned Task . . . 10
  - 2.3.2 Implementation . . . 10
  - 2.3.3 Learning Outcomes . . . 10
- 2.4 elm-array-view . . . 10
  - 2.4.1 Assigned Task . . . 10
  - 2.4.2 Implementation . . . 11
  - 2.4.3 Learning Outcomes . . . 12

3 Source Codes and Documents **13**
- 3.1 Source Codes . . . 13
- 3.2 Documents . . . 13

4 Conclusion **14**

1 Introduction

1.1 Learning Objectives

- To get familiarized with the language of Elm and in turn getting a good idea of functional programming.
- To get an idea of how industry-standard code should be made, in order to make it easy to read and modify.
- To make dynamic code which can automatically adjust its parameters depending on the users input.
- Using inference obtained from existing packages to make our own visualization package with which a given data structure can be visualized.

1.2 About Internship

As an intern with SERC, I was given the responsibility of contributing to a visualization package which will be utilized in the algodynamics page. This would help in an easier implementation with the added possibility to include more features or handle different events.

During my internship with the research center, I was able to grasp a detailed understanding of the functional programming language, Elm. With the help of various learning resources like Beginning $\| > Elm$ and An Introduction to Elm.

By utilizing the language of Elm, I was able to realize the importance of functional programming and how it can be a much better option than compared to the extremely popular language of JavaScript as it has features such as Static Typing and Immutable Data.

I was able to get a good understanding on how to implement existing packages and correctly format Elm code through a few projects which will be explained about in detail in the later sections.

1.3 Installation and Integration

A detailed description on the installation of elm can be found here

To check whether your installation was successful, you can enter `elm --version` in your terminal. If elm was installed properly, you should find the version number being displayed in the terminal.

Common commands used :

- `elm init` : Initializes an elm project with an `elm.json` file and `src` directory. `elm.json` is used to describe the dependencies and `src` is used to store Elm files.
- `elm reactor` : Builds elm project and starts a server at http://localhost:8000/
- `elm make` : Compiles Elm code to HTML or JavaScript
- `elm install` : Used to install packages from package.elm-lang.org. Installed packages are added as dependencies in the `elm.json` file.

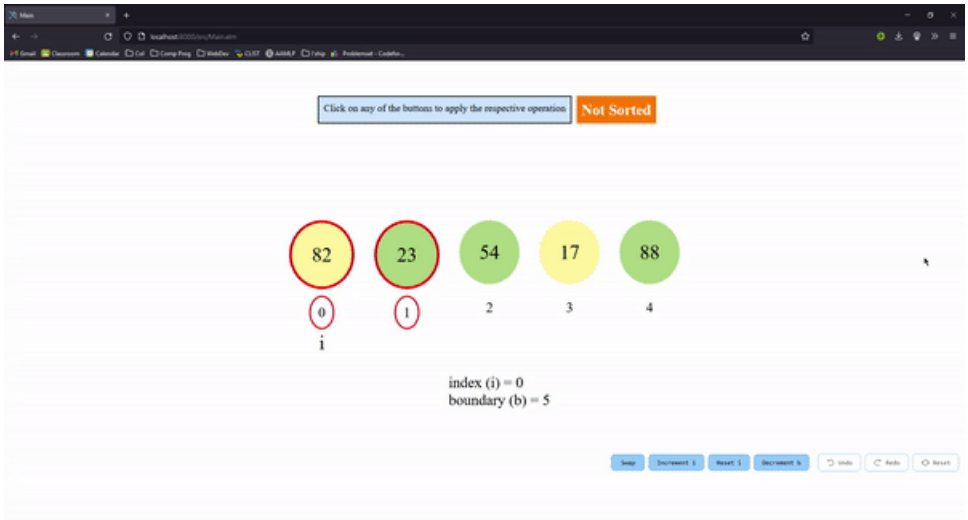
2 Projects

2.1 elm-bblsort

2.1.1 Assigned Task

Replicate the implementation of the bubble sort exercise present in the Algodynamics site.

2.1.2 Implementation



- Unique messages are displayed depending on the users chosen operation. For example, if the user is deviating from the bubble sort algorithm, a message is displayed to warn the user.
- Elements in the correct position are highlighted in green, giving a clear indication on which elements are correctly placed.
- Transitions were added too, to make it more visually appealing.
- Utilized Core Package to include the Undo, Redo and Reset features.

2.1.3 Learning Outcomes

- Got a good understanding of how elm code should be structured.
- Learnt to utilize other packages in the project, obtained from elm packages site.
- Familiarized myself with the significance of model-view-update architecture in Elm.

2.2 elm-calci

2.2.1 Assigned Task

Implement a simple calculator using Elm.

2.2.2 Implementation

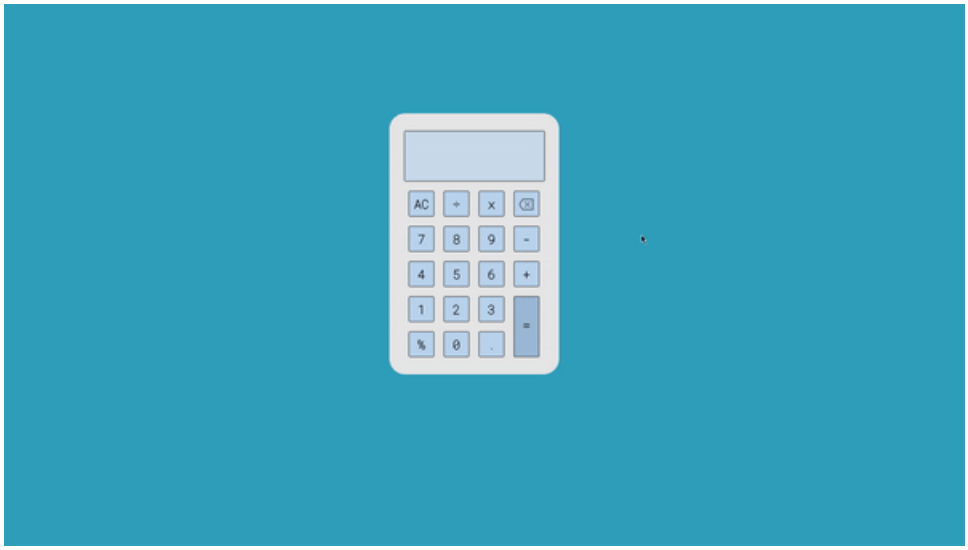


Figure 1: Implementation without template code

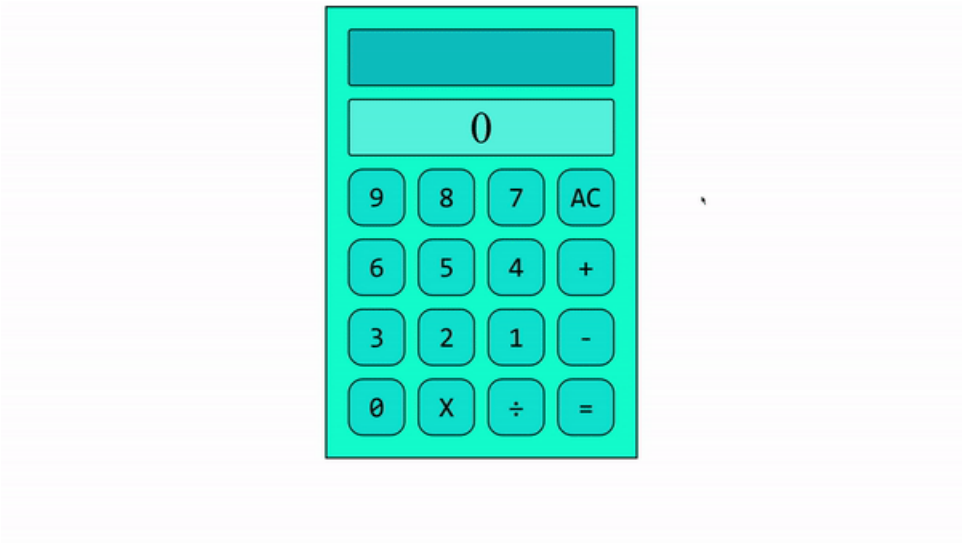


Figure 2: Implementation with template code

- In the implementation without the template code, the display shows two values - one of which shows the input entered by the user, and the other shows the history of calculations carried out.
- But it had a drawback - the operations carried out by the user was not in accordance with BODMAS rule. So there was a scope of inaccuracy.
- This drawback was rectified with the help of the second implementation.
- It was also ensured that the button SVG's are generated in a single SVG canvas through the aid of a recursive function, which will come in handy in the next project.
- Attributes to modify the button styling, dimensions and arrangement have also been added.

2.2.3 Learning Outcomes

- Grasped a good understanding of utilizing SVG with Elm, to elements with user-desired configuration.
- Learnt how to make custom attributes, which can be assigned a default value if the user chooses not to enter their own value for the same.

2.3 elm-dagre

2.3.1 Assigned Task

Document the working of elm-dagre package and showcasing the significance of some of the modules present in the package.

2.3.2 Implementation

Descriptions of the attributes and modules can be found in this markdown file.

2.3.3 Learning Outcomes

- Was able to get an idea about the various attributes and drawers used to generate the visualization for the user specified graph.
- Got an understanding on how an elm package is made, which is what is to be done in the next task.

2.4 elm-array-view

2.4.1 Assigned Task

Create code which generates the visualization for a user-specified array and publish a package on the same.

2.4.2 Implementation

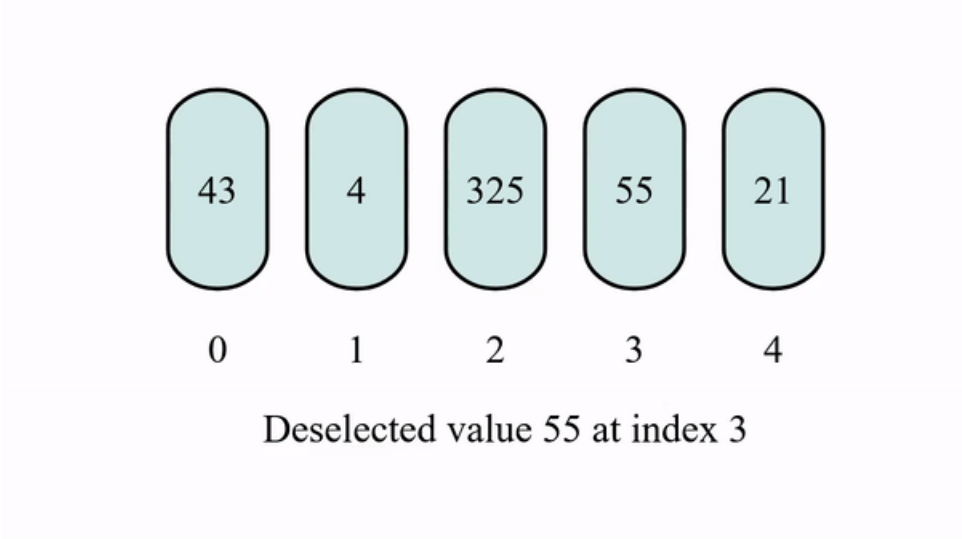


Figure 3: Implementation without elm-dagre

Figure 4: Implementation with elm-dagre

- A user specified array is visualized with the help of this module.
- Through added user attributes, the dimensions, shape and positioning of index labels can be modified.
- The first implementation has a drawback of not having a wrap feature and having static index label positions. Also, the array containers are not individually configurable.
- These drawbacks were taken care of in the second implementation, in which the elm-dagre package is utilized.
- A custom drawer is created in order to avoid any confusion with the graph attributes.

- Positioning of index labels can be set at any position by specifying the desired X and Y co-ordinates.
- Additional features such as setting a limit of the number of elements that can be present in a single row can also be added by the user by making minor tweaks to the drawer function.

2.4.3 Learning Outcomes

- With the help of what we've learnt from the previous projects, we were able to implement this array visualization function.
- Learnt about the process involved in deploying an elm package and how one can properly document it.
- Grasped valuable insight on making simple and well structured code so that one can easily make tweaks to it if they need to.

3 Source Codes and Documents

3.1 Source Codes

elm-bblsort	github.com/kmvolv/bbl-sort
elm-calci	Without template — With template
elm-dagre	Markdown File
elm-array-view	Without elm-dagre — With elm-dagre

3.2 Documents

Work Log
Progress Report

4 Conclusion

Through my internship at SERC, I was able to contribute to making an array visualization package in the language of Elm. Through this package, the user can obtain the visualization of an array in SVG format. The user is given liberty to add their own attributes to the module, and decide how that attribute affects the rendered output.

With the help of the projects carried out preceding the package, I was able to get a good understanding on how a package can be built from the ground up as well as to create custom attributes to make user-desired modifications to the output, as stated in the aforementioned paragraph.

Overall, I found this internship experience to be very worthwhile and I'm sure that I will be utilizing the skills and insights that I have learnt here in later stages of my career.