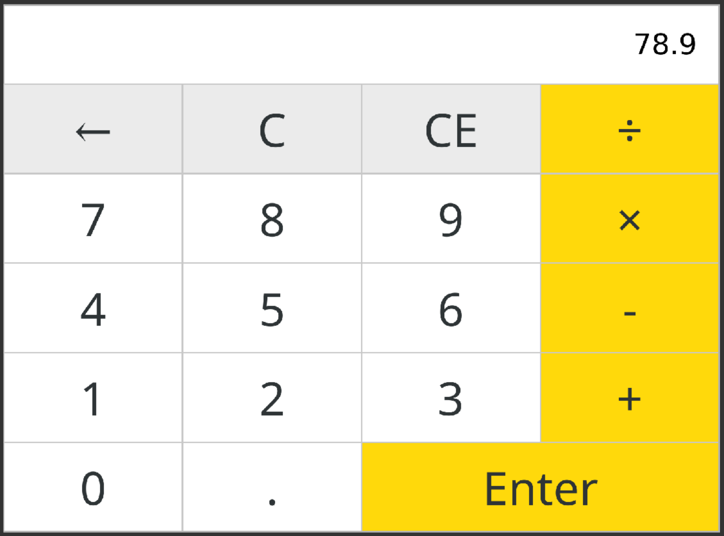
Elm 0.19

Learn Elm

by example



Code a Calculator using Elm from scratch

Ryan Frazier

Contents

List of Figures

1 Introduction

1.1 Who am I?

I have been working as a software engineer for about 4 years and I like to program mostly for the web. In my day to day work I use Vue.js a lot (still trying to convince my co-workers to try Elm, hence this book).

I have experience writing backends in Node.js, Python, and Java. I also love static site generators like Hugo and 11ty.

I love to learn new things and try new things out. On a whim I decided to try Elm because it looked so different from what I was used to. I tried it and I was pleasantly surprised. If I am going to write something complex in the browser Elm is my first choice.

1.2 Learn by doing

Whenever I learn something new, I learn best by actually doing it. If you want to learn a new computer language pick a project and just go for it.

This book is an attempt to guide a beginner through building something non-trivial in Elm. It will be very light on the theory and focus on building stuff.

1.3 Who is this book for?

I wrote this book because I found it difficult to recommend tutorials to those wanting to start using Elm. Beginners might feel intimidated by the new syntax and the non-component way of doing things. If you are coming from React or Vue things will look a bit different. There are no components. Only functions.

After dabbling in Elm for a while I knew I wanted to use it in a larger project but I didn't know where to get started. There seemed like I had to learn so much it was easy to get discouraged. I kept coming back to Elm hoping things would click.

This is the book I wish I had when starting to learn Elm.

Hopefully this will save you time and help you get past the rough patches when starting out with Elm.

1.4 What will be covered?

- Elm's type system
- CSS with external style sheet
- elm-live development server
- custom keyboard input events
- key combo events (like alt-shift-delete)
- testing with elm-program-test and elm-test
- Browser.sandbox vs Browser.element vs Browser.document
- refactoring an Elm application
- deployment the application with Netlify

1.5 How to use this book and provided resources

Provided with this book are the following:

- a git bundle of the repository
- diffs of each chapter
- a link to the ellie-app version of the code

1.5.1 How to use the git bundle

I have provided the git repository for you to download as a git bundle. You can clone this bundle just like you would a repository from GitHub or GitLab.

Once you download the bundle, go to the directory you have the bundle in and run the clone command.

```
git clone elm-calculator.bundle
```

And you now have a fresh clone of the repository with all my commits and history for this project.

I have tagged each commit `v0.1`, `v0.2`, and so on. They correspond to each chapter of the book. I will reference each tag at the beginning of each chapter.

```
git checkout v0.1
```

Using VSCode you can see a good diff comparison between the different version of the app. See the accompanying video.

1.5.2 What is ellie app?

Ellie app is an online environment to code in Elm online. There is no setup involved. It is similar to CodePen or JSbin.

Go to https://ellie-app.com.

1.6 What are we going to build?

We are going to build a calculator. Since they are easier to program, we are going to build a Reverse Polish Notation (RPN) calculator.

Here is what it will look like when it is done.

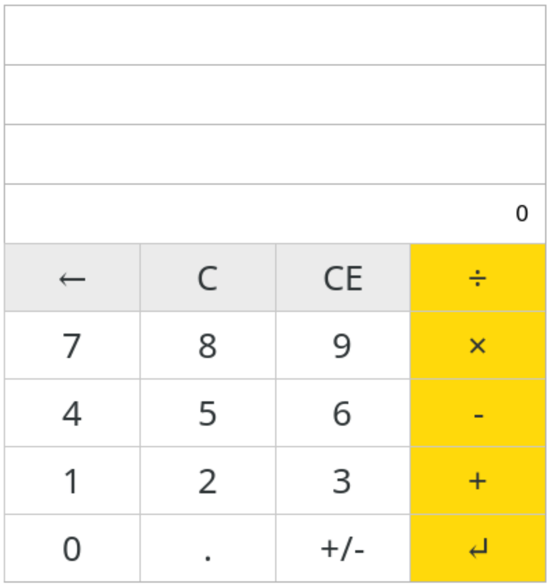


Figure 1: RPN Calculator

Go to https://elm-calculator.netlify.com for a demo of the finished calculator.

An RPN Calculator is easier to program because all operations work on the stack. Let's briefly explore the stack and how it works with our calculator.

1.7 The Stack

A stack is a data structure that has two operations: push and pop. Similar to a stack of plates, you can push onto the stack by placing things on top. Likewise you take things off the stack by popping off the top. Another name for this is LIFO (last in first out). See fig. 2.

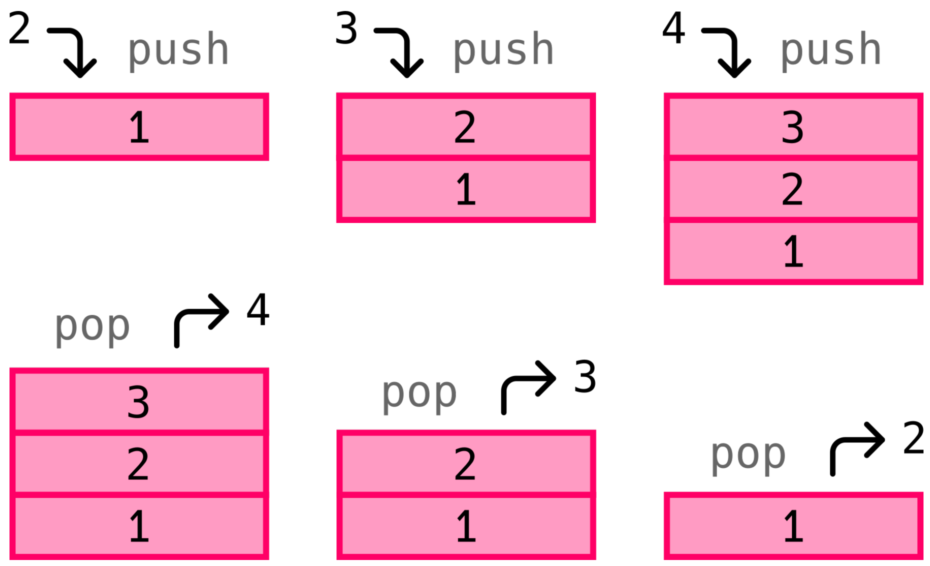


Figure 2: The Stack

For our calculator, we push numbers onto the stack. To pop items off we perform operations on the numbers of the stack. The operators "+", "-", "÷", and "×" need to operate on two numbers, so these operators will pop 2 numbers off the stack, operate on them and then push the result back onto the stack.

For example we push "1" and then "2" onto the stack. Then press the "+" operator. This pops "1" and "2" off the stack, operates on them and then pushes "3" back onto the stack. See fig. 3.

BLANK PAGE

2 Setup the project

Like I said in the previous chapter, I have provided the git repository for you to download as a git bundle.

Make sure you clone the repository and then checkout the version of the app for this chapter.

```
git clone elm-calculator.bundle
cd elm-calculator
git checkout v0.1
```

- ***browse:*** `git checkout v0.1`
- ***ellie:*** https://ellie-app.com/72nScWrSMfVa1

2.1 Setup your editor

See the video provided with the course to walk through setting up VSCode. You can skip this step if you just want to develop inside ellie-app.

For more editor choices see this page.

2.2 Initialize elm and npm

Now that our editor is setup we can write some code.

Create a new directory and open a terminal in that directory.

```
mkdir elm-calculator
cd elm-calculator

elm init
npm init
npm install --save-dev elm-live
```

Now create the following files: `index.html`, `style.css`, and an Elm file `src/Main.elm`.

```
touch index.html style.css src/Main.elm
```

You should have a file structure that looks like this.

```
elm-calculator
├── elm.json
├── index.html
├── package.json
├── src
│   └── Main.elm
└── style.css
```

2.3 The Html

The important thing here is that the Elm is loaded properly. The Elm compiler will compile our Elm program to an `elm.js` file. We need to load that file and initialize it in the `index.html`.

We can also link a CSS file in the header as well.

```html
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style.css">
    <script src="elm.js"></script>
    <title>Elm Calculator</title>
</head>

<body>
    <main></main>
    <script>
        var app = Elm.Main.init({ node: document.querySelector('main') })
    </script>
</body>
```

```
</html>
```

2.4 The CSS

We'll add a little bit of style just to convince us that it is working.

```css
/* style.css */
body {
    font-family: sans-serif;
    text-align: center;
    padding-top: 3em;
    font-style: italic;
}
```

2.5 A Minimal Elm Program

And here is the most minimal elm program. Every Elm program needs a model, update, and view function.

```elm
-- src/Main.elm
module Main exposing (main)

-- import the things we will be using
import Browser
import Html exposing (Html, h1, text)


-- an empty model for now
type alias Model =
    {}

-- our model always needs an initial state
initialModel : Model
initialModel =
    {}
```

```elm
-- since the user can't do anything yet
-- we'll define a no-operation message
type Msg
    = NoOp


update : Msg -> Model -> Model
update msg model =
    case msg of
        NoOp ->
            model


view : Model -> Html Msg
view model =
    h1 [] [ text "Hello Elm!" ]


main : Program () Model Msg
main =
    Browser.sandbox
        { init = initialModel
        , view = view
        , update = update
        }
```

2.6 Run the project

```
npx elm-live src/Main.elm --hot --open -- --output=elm.js
```

npx ships with node.js and npm. It allows us to run programs that are installed locally by npm in the project directory. The alternative would be to install elm-live globally on your machine with npm `install --global elm-live` and use elm-live without npx.

`--hot` tells elm-live to inject any changes into the app without requiring a hard browser refresh.

`--open` tells elm-live to open the browser when the program is done being compiled.

Anything after the `--` are flags that are passed directly to the elm compiler. In this case we want elm to output a JavaScript file that we can include in our `index.html`.

If all goes well you should see the following output.

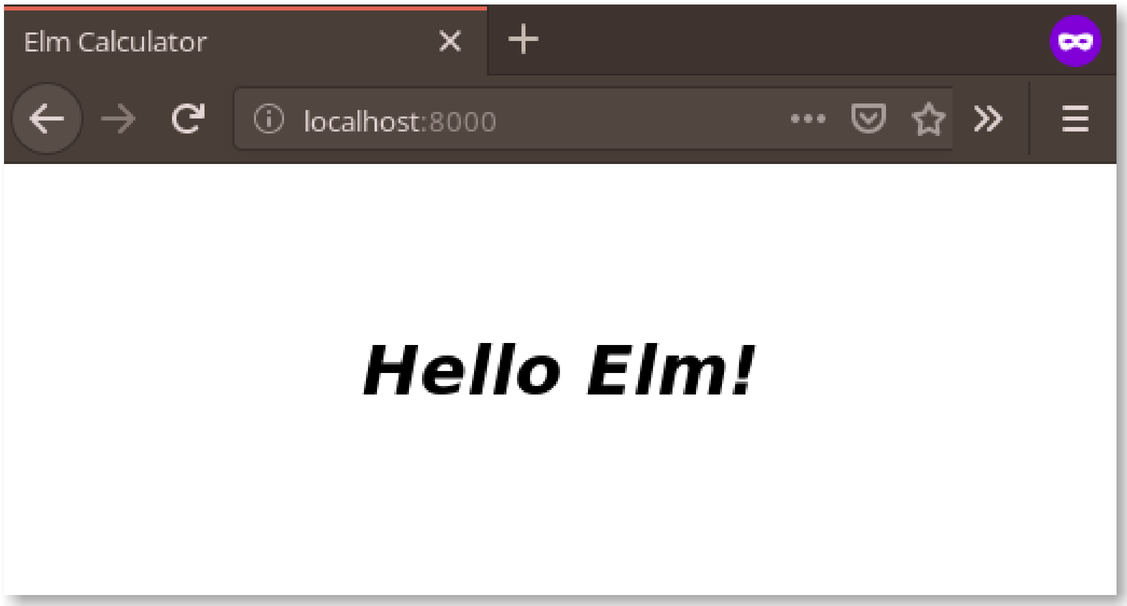


Figure 11: Hello Elm

For more information about the options to elm-live see the documentation here.

2.7 Setup Shortcut

I have created a project template on GitHub that does this whole setup for you at https://github.com/pianomanfrazier/elm-starter.

If you have a GitHub account you can simply click the "Use this template" button and GitHub will make a repo in your account for you.