CS4215 Project Report

PROGRAMMING LANGUAGE IMPLEMENTATION

Mini Project

"Minions' Chatting Room"

Submitted by

A0174341L Yahui Song

Under the guidance of

Prof. Chin Wei Ngan

&

Prof. Razvan Voicu

School of Computing

National University of Singapore

Singapore

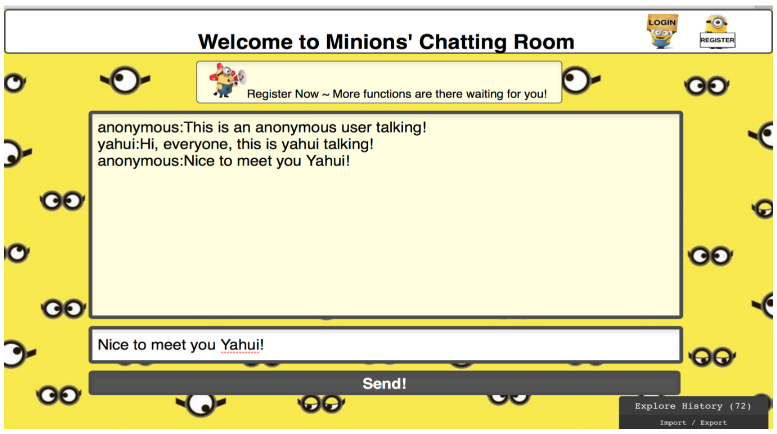
Semester 2 2017/2018

1 Introduction

Programming languages are important fundamental artifacts for building large and complex software. The purpose of this mini project is to do a systematic exploration in some specialized domains.

First of all, this is the repository of this project on github:
`https://github.com/songyahui/CS4215-mini-project.git`

In my project, I used functional programming language Elm to construct a web application. This application is a chatting room with the theme of the minions, called "Minions' Chatting Room". My motivation of doing this project is that Elm is an advanced language which specified to construct web applications, it has its own wonderful features both on software designing and coding process. It is necessary and useful to try it out. And to build a corresponding server and database is also essential for a web application nowadays in general. The main function of this application is to transfer messages between different users using broadcast. Besides, it also has several extra functions such as log in, register, checking online users' list and so on. As it showed in Figure 1, this is the home page of this application:


Welcome to Minions' Chatting Room
LOGIN
REGISTER
Register Now ~ More functions are there waiting for you!
Send!
Explore History (72)
Import / Export
anonymous:This is an anonymous user talking!
yahui:Hi, everyone, this is yahui talking!
anonymous:Nice to meet you Yahui!

Nice to meet you Yahui!

Figure 1: Home Page

What's more. There are several other pages such as login, register, logout and so on. And the jump between those pages is fluent. There are also some beautifully designed icons can be seen in those pages.

2 Implementation

In this section, I will explain the implementation of this application in detail. Server, clients and database construct this application. Technically speaking, websocket and sql query are the foundation of building this application.

2.1 Server

The server in this application is written in node.js. As it showed in Listing 1, the server listens to a port which is known by clients, and they can transfer messages through ip address and port number. This piece of code is just a simple example of web socket usage in node.js. In this project, there are 3 kinds of type of messages can be received by server: MESSAGE, LOGIN, REGISTER. So it also has other functions to deal with different types of messages. Which can be easily checked in the source code folder under root "~\server\char-server.js". And use command "node char-server.js" to set up the server.

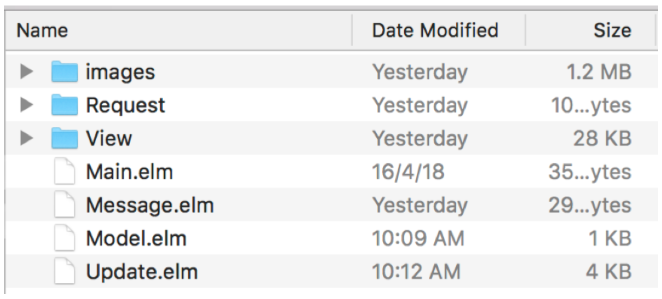
Listing 1: Server.js

```
var portNumber = 1234;

app.listen(portNumber, function() {
  console.log(`Listening on port ${portNumber}`);
});
app.ws('/hello', function(websocket, request) {
  console.log('A client connected!');

  websocket.on('message', function(message) {
    console.log(`A client sent a message: ${message}`);
    websocket.send('Hello, world!');
  });
});
```

2.2 Client

The client is written in Elm. Typically, Elm's architecture has four parts: View, Update, Message and Model. To make it clear, I divide all the source code correspondingly, as it showed in Figure 2. "images" folder saves all the pictures used in UI. "Request" folder saves a helper to connect to server.

"View" folder has several files which correspond to different pages. In this way, we can separate all the CSS and HTML code used for layout and all the



Name	Date Modified	Size
▶ images	Yesterday	1.2 MB
▶ Request	Yesterday	10...ytes
▶ View	Yesterday	28 KB
Main.elm	16/4/18	35...ytes
Message.elm	Yesterday	29...ytes
Model.elm	10:09 AM	1 KB
Update.elm	10:12 AM	4 KB

Figure 2: Elm root folder

process code used for the functional aims.

Designing of the Model is very important. It should contains all the variables which might be changed during the whole process. As it showed in Listing 2. “currentRoute” represents the current page which could be “login-page” or “home-page” etc. The value of “status” could be: Anonymous, Registered, Login-InfoErr etc. which represents different status of the user currently.

Listing 2: Model

```
type alias Model =
    { currentRoute : Navigation.Location
    , userId : Maybe String
    , userName : Maybe String
    , userPassword : Maybe String
    , userPassword_again : Maybe String
    , input_message : Maybe String
    , show_messages : List String
    , history_message : List String
    , online_user_list : List String
    , status: Status
    }
```

2.3 Database

I use MYSQL as my database. There are mainly 2 tables, “users” and “records”. First one is used to save all the information of users while second one is used to save all the messages send by login users.

3 Future work

I think currently there are two aspects of future work I can do. Firstly, Encrypting data. Even though the network protocol has already guaranteed the safety of all the data transferred between server and client, there are still some very important information should be kept safer like password. Secondly, UI design. Even though Elm used the API of HTML and CSS derectly, sometimes I still encounter some difficulties on the layout, and also the UI design will give the first impression of this application to users, it always could be improved!

4 Conclusion

During the whole process of programming this project. I mainly used two programming languages: Node.js and Elm which in some degree can be represent imperative programming and functional programming. The intuition feeling after this project is that even thought Node.js is easy to program, it kept throwing our exceptions and had lots of crash while Elm never had run-time errors.

Setting up a web application, Elm is definitely on the right track. I think that if we are trying to figure out what to use for our next project, we should give Elm a try. We can even use it with existing projects due to the interoperability with JavaScript. It has a lot of benefits and lasting positive effects on a team and code base.

5 Reference

1. Elm and Express: A Simple Client-Server Implementation
   `https://spin.atomicobject.com`

2. An Introduction to Elm
   `https://guide.elm-lang.org`

3. Realword SPA Example
   `https://github.com/gothinkster/realworld`