Thermometer App

High Level Design

1.0

November 17, 2013

Ray Yi

Software Engineer

Prepared for

WUBS UI Thermometer Assignment

**Table of Contents**

1. General Description 1

2. Hardware 1

3. Software 1

3.1. node.js and express framework 1

3.2. Twitter bootstrap 1

3.3. JQuery and angular.js 1

3.4. Jasmine 1

4. Sequence Diagram 2

4.1. Main View 2

4.1.1. Diagram script: 2

4.2. Setting View 3

4.2.1. Diagram script: 3

5. Component Design 5

5.1. Default page 5

5.2. Main View 5

5.3. Main Controller 5

5.4. Setting View 5

5.5. Setting Controller 5

5.6. Setting Model 5

5.7. Temperature Service 5

5.8. Temperature API 5

# General Description

This document describes high level design for the project Thermometer App.

# Hardware

This app is web-based application, it will run in any browser with HTML 5 support.

We use node.js as www server, it can run on Windows/Mac/Linux.

# Software

## node.js and express framework

I use node.js as www server, and also create temperature API by using express framework.

## Twitter bootstrap

I use twitter bootstrap for UI styling.

## JQuery and angular.js

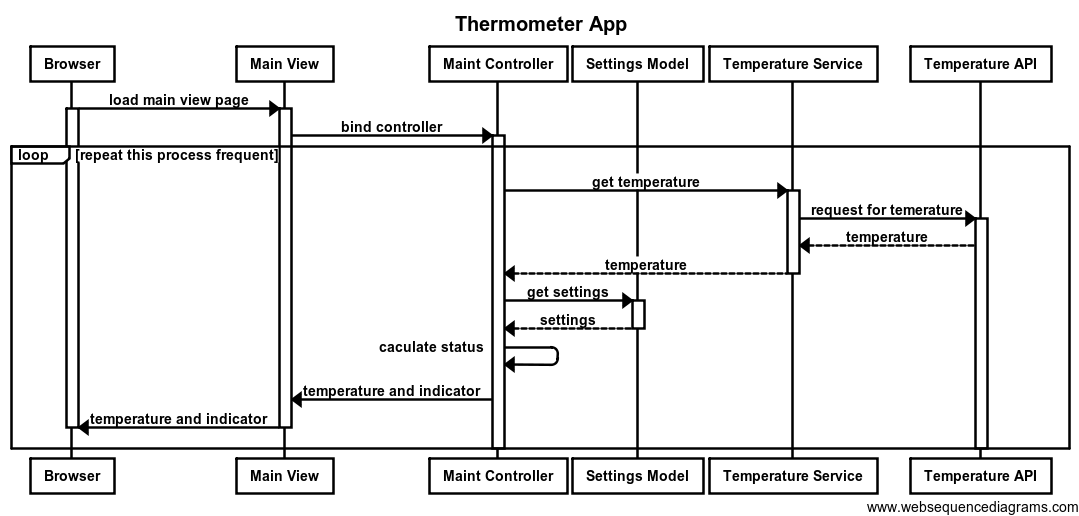
I use JQuery and angular.js to develop this single-page-application.

## Jasmine

I use Jasmine test framework for unit test.

# Sequence Diagram

## Main View



### Diagram script:

title Thermometer App

participant Browser as br

participant "Main View" as mv

participant "Maint Controller" as mc

participant "Settings Model" as sm

participant "Temperature Service" as ts

participant "Temperature API" as api

br->mv: load main view page

activate br

activate mv

mv->mc: bind controller

activate mc

loop repeat this process frequent

mc->ts: get temperature

activate ts

ts->api: request for temerature

activate api

api-->ts: temperature

ts-->mc: temperature

deactivate ts

mc->sm: get settings

activate sm

sm-->mc: settings

deactivate sm

mc->mc: caculate status

mc->mv: temperature and indicator

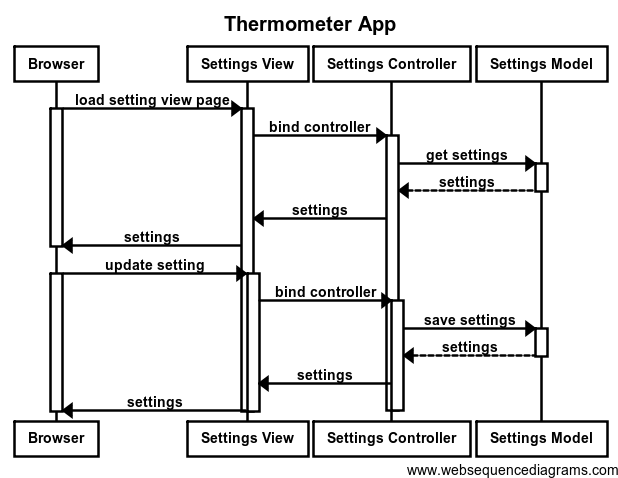
mv->br: temperature and indicator

end

deactivate mv

deactivate br

## Setting View



### Diagram script:

title Thermometer App

participant Browser as br

participant "Settings View" as sv

participant "Settings Controller" as sc

participant "Settings Model" as sm

br->sv: load setting view page

activate br

activate sv

sv->sc: bind controller

activate sc

sc->sm: get settings

activate sm

sm-->sc: settings

deactivate sm

sc->sv: settings

sv->br: settings

deactivate mv

deactivate br

br->sv: update setting

activate br

activate sv

sv->sc: bind controller

activate sc

sc->sm: save settings

activate sm

sm-->sc: settings

deactivate sm

sc->sv: settings

sv->br: settings

deactivate mv

deactivate br

# Component Design

## Default page

This is default html page in public folder, which is auto loaded into browser. It is bind to angular app. It contains place holder for partial html for main view and setting view.

## Main View

This is partial html file, it binds to main controller to get temperature and indicators.

## Main Controller

This controller takes below process:

1. call method ‘getTemperature’ to get temperature from temperature API through ‘temperature service’.
2. in the call back function, it calls method ‘updateStatus’ to update temperature and indicators for main view.
3. use ‘setTimeout’ method to run this method again.

Step #2 and #3 need use values in Setting Model as parameter.

## Setting View

This is partial html file, it binds to setting controller to get/update settings.

## Setting Controller

It connects setting view and setting model.

## Setting Model

It is json object, it contain all attributes used in setting page.

## Temperature Service

It is angular service, it call Temperature API to get temperature and return promise object.

## Temperature API

This is web method return temperature to client side.