Faculty of Science, Engineering and Technology

Interface Design and Development

Pass Task 3: AngularJS-Expression & Conditional Directives

Overview

By using expression and conditional directive in AngularJS, it makes creating programs that dynamically respond to user input possible. In this task you will create a small web application that uses conditions and filter to dynamically output custom messages to users and to compute the body mass index (BMI).

Purpose:	Install and test the framework needed to get started and learn to use expressions and conditional directives to respond to user actions.
Task:	Create a web app that tests a user's name and a web app that computes BMI to displays a custom message
Time:	This task should be completed in your lab class and submitted for feedback before the start of week 5.
Resources:	▪ Lecture notes #3 ▪ AngularJS https://angularjs.org/

Submission Details

You must submit the following files to Blackboard:

- Name Tester & BMI calculator source code (nametester.html & bmi.html).
- Screenshot of the BMI web app.

Make sure that your task has the following in your submission:

- The Name Tester web application is HTML5 compliant.
- Demonstrates understanding in using the AngularJS framework.
- Demonstrates use of AngularJS conditional directives.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Component 1: Name Tester Web App with AngularJS

Instructions

The first task includes the steps needed for you to install the framework you will need in this unit. You will then use the AngularJS framework to create the 'Hello World' web page.

1. Download and install the framework you need to get started. Ensure that you have:
   - Installed AngularJS (https://angularjs.org/) – **Version 1.7.***

Note: Don't Try the **New Angular / Angular 2.0** onwards, it is a totally different framework from AngularJS

You should have the following files stored in appropriate folders

```
framework/
├── css/
│   ├── bootstrap.min.css
│   ├── bootstrap.min.css.map       ← map is used for debugging
│   ├── bootstrap-theme.min.css
│   └── bootstrap-theme.min.css.map
├── js/
│   ├── angular.min.js              ← can remove version number
│   ├── bootstrap.min.js
│   └── jquery.min.js               ← can remove version number
└── fonts/
    ├── glyphicons-halflings-regular.eot
    ├── glyphicons-halflings-regular.svg
    ├── glyphicons-halflings-regular.ttf
    ├── glyphicons-halflings-regular.woff
    └── glyphicons-halflings-regular.woff2
```

Note: You may also want to download scripts to enable legacy versions of IE to fully support HTML5 and CSS3, and place them in the js folder. These are html5shiv https://cdnjs.com/libraries/html5shiv and respond https://cdnjs.com/libraries/respond.js/

Tip: You can also use direct link to specific versions of the scripts in your HTML file.

```
<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
<!--[if lt IE 9]>
  <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
  <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
<![endif]-->
```

2. If you don't already have one, make a directory (i.e., a 'folder') to store your framework (e.g. Documents/cos30043/lab03). On a Swinburne computer you may wish to use a directory on your student drive or a USB storage device.

Note: Preferably replicate a copy of the framework files for each lab.

3. Save the file as nametester.html in your lab03 directory.

4. Start the web application code with the template for AngularJS found in lecture 03.

5. Implement a web application with the following logic:
   - It reads a name from the user, and displays back a message.
   - Check if the name entered is your name.
   - If the name is your name, output the message 'Awesome name!'
   - Otherwise output the 'not my name' message.

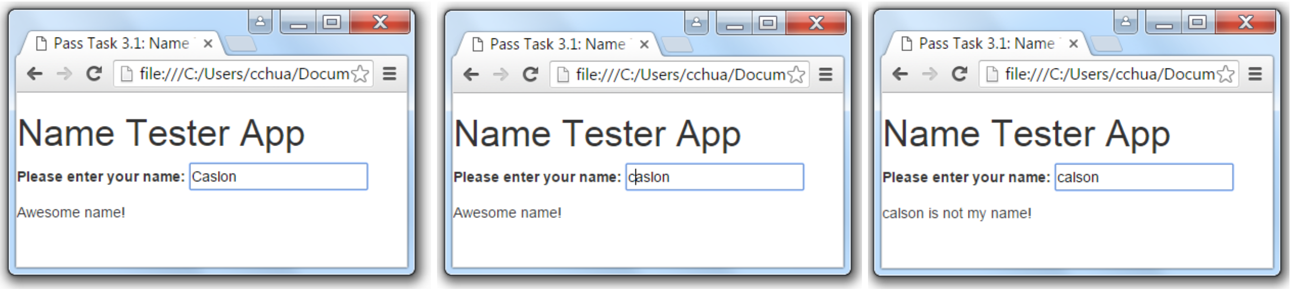
```
Web App: nametester.html

Uses: AngularJS
--------------------------------
- Model:
- strName (which stores a String value)
--------------------------------
- Steps:
1: Assign strName using ng-model with the prompt: 'Please
    enter your name:'
2: ng-show (if)"strName == '–add your name here–'"
3:         Output 'Awesome name!'
4: ng-hide (else)"strName == '–add your name here–'"
5:         Output strName, ' is a not my name'
```

Tip: Remember to use == as the conditional operator, = is an assignment operator.

Note: Conditional operator is case sensitive. For example, Caslon is not equal to caslon

Hint: Use the lowercase filter to convert the value of a variable into lowercase before you compare.


Pass Task 3.1: Name
file:///C:/Users/cchua/Docum
Pass Task 3.1: Name
file:///C:/Users/cchua/Docum
Pass Task 3.1: Name
file:///C:/Users/cchua/Docum

Name Tester App

Please enter your name: Caslon

Awesome name!

Name Tester App

Please enter your name: caslon

Awesome name!

Name Tester App

Please enter your name: calson

calson is not my name!

Figure 1: Screenshot of the web app with no Bootstrap mark up

Component 2: Health Web App

Instructions

Without AngularJS, writing a web application with computational interactive can be quite tedious. Using JavaScript will require a number of listeners to be hooked up to the input form elements in order to interactively update the view with the updated computed values.

With AngularJS, the task is straight forward using model and expressions. And custom message can be implemented using conditional directives, such as ng-if, ng-show/ng-hide and ng-switch.

1. Open Brackets and save the blank file as bmi.html in your lab03 directory.

2. Start the web application code with the template for AngularJS found in lecture 03.

3. Implement a web application with the following logic:
   - It reads 2 inputs, namely: weight in kilograms, and height in centimetres.
   - Compute and show the calculated BMI. The formula is

     BMI = bodyweight in kilograms divided by height in meters squared
   - Display the appropriate custom message whether the calculated BMI is ‘underweight’, ‘normal’. ‘overweight’ or ‘obese' based on the following BMI cut off points

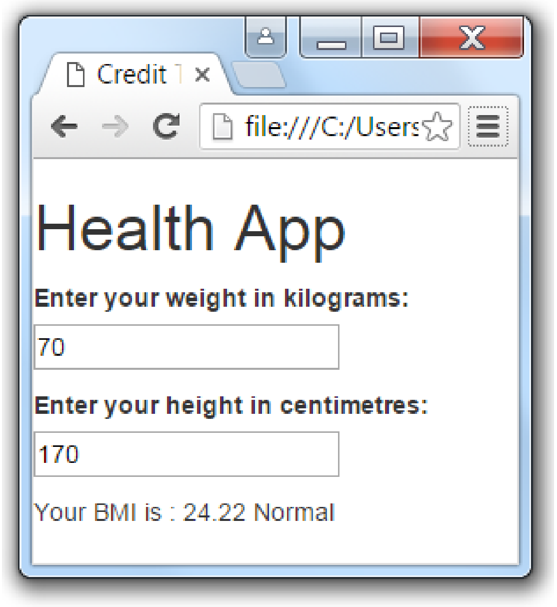
Underweight:	< 18.5
Normal weight:	18.5 to 24.99
Overweight:	25 to 29.99
Obese:	≥ 30

```
Web App: bmi.html
________________
Uses: AngularJS
--------------------------------
- Model:
- numBMI (computed BMI)
- numWt  (weight in kilograms)
- numHt  (height in centimetres)
--------------------------------
- Steps:
1: Initialise numWt and numHt to 0 using ng-init
2: Assign numWt using ng-model with the prompt: 'Enter your
    weight in kilograms:'
3: Assign numHt using ng-model with the prompt: 'Enter your
    height in centimetres:'
4: Show with 2 decimal places 'Your BMI is: '-add (assign
    and compute BMI)
5: Using ng-if compare BMI with the range
6: Output custom message
7: Repeat (5-6) 3 more times using the subsequent range
```

Tip: Remember height is in centimetre while formula solves height in meter.

Hint: ng-init can be place in the open body tag and use the assignment operator expression. For example, `{{answer = 3 + 4}}` will assign 7 to the variable answer, at the same time display the number 7.

Note: Form element is not used in this task. You may want to use Bootstrap.



Health App

Enter your weight in kilograms:

70

Enter your height in centimetres:

170

Your BMI is : 24.22 Normal

Figure 1: Screenshot of the web app with no Bootstrap mark up

Now that the task is complete you can submit it for assessment, which will help prepare it for your portfolio.