Lab – AngularJS

May 2015

Document Control

Version	1
Created by	Namrata Marathe
Reviewed by	Chand Ali
Date	January 2015
Earlier versions	January 2015

Table of Contents

1. Lab 1 3
2. Lab 2 4
3. Lab 3 6
4. Lab 4 6
5. Lab 5 8

1. Lab 1

- Declare a variable 'myname' and json object called as 'contact' in controller function 'ContactController' and access it in view.

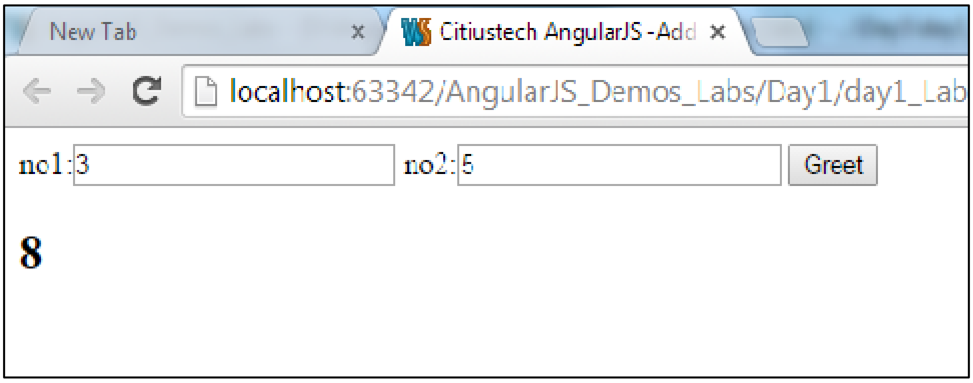
```
myname = "Arjun"
contact = { name:"Namrata Marathe", price:3.00,
            description:'This is fine quality costliest gem',
            canPurchase:false,
          soldOut:true,}
```

lab 1:1 scope controller lab

- Repeat the same example above and fill in the proper directives.

lab1:2 scope json controller lab

- Take two numbers 'no1' and 'no2' from end user, write add function inside controller to add those two numbers and store the result in 'no3'.Display 'no3' in view.

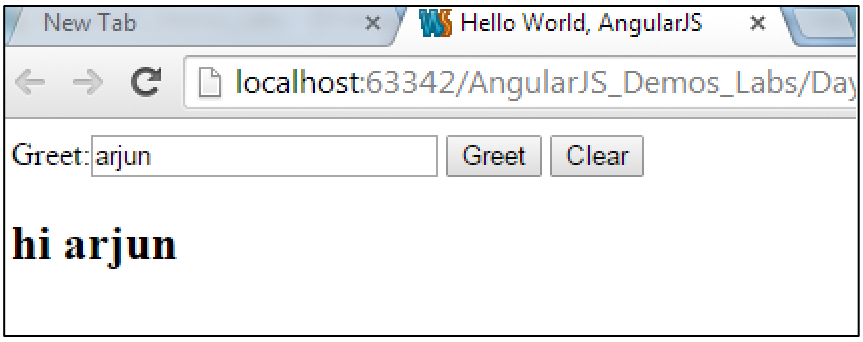

New Tab
Citiustech AngularJS -Add
localhost:63342/AngularJS_Demos_Labs/Day1/day1_Lab
Greet

no1: 3 no2: 5

8

Lab1:3 add scope controller lab

- Take name from enduser in 'myname' variable and add two buttons 'greet' and 'clear'.
- When end-user clicks on 'greet' button it should greet user with his name entered in textbox. On clicking 'clear' button only value entered in 'textbox' should be cleared.



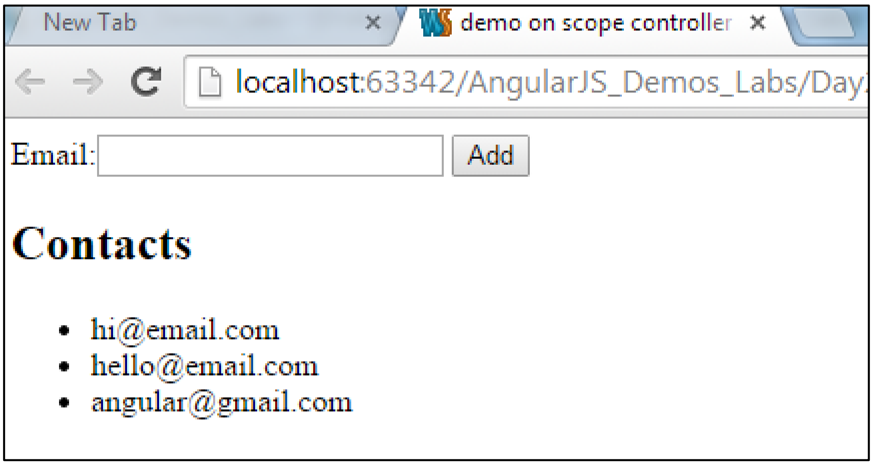
Greet: arjun

hi arjun

Lab1:4 ngclick controller lab

2. Lab 2

- Take email id from end-user in a textbox and 'add' button. As end-user enters emailid it should get listed below as follows: (use array)



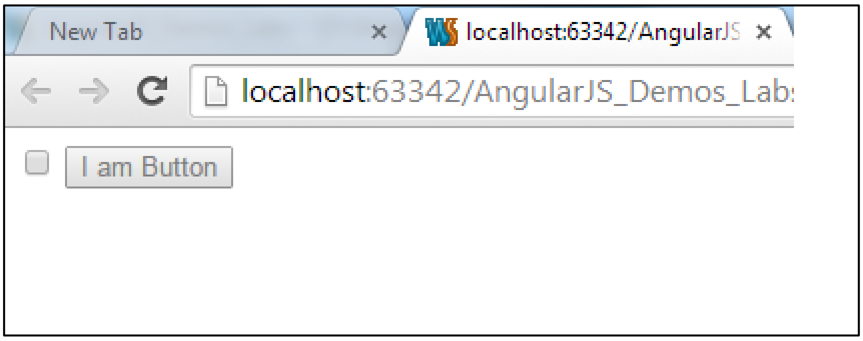
Email:

Contacts

- hi@email.com
- hello@email.com
- angular@gmail.com

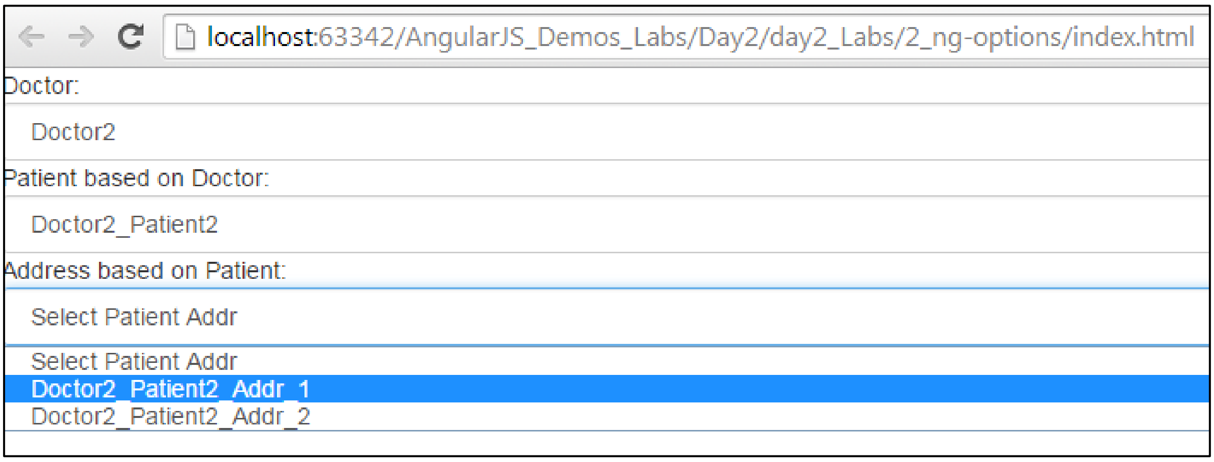
[Lab2:ng_repeat_lab]()

- Create a checkbox and a button. when you check the checkbox the button should be enabled, on unchecking it should be disabled


New Tab
localhost:63342/AngularJS
localhost:63342/AngularJS_Demos_Lab
□
I am Button

[Lab2:button_disable_chkbox]()

- Lab on ng-options
- Create 3 drop-down boxes as shown below.
- First drop-down box should contain list of doctors
- Second drop-down should contain list of patients
- Third- drop-down should contain list of addresses
- As you select doctor from first drop-down, correspondingly second-drop down should be populated with list of patients belonging to doctor selected in first drop-down
- As you select patient from second-drop-down correspondingly third drop-down should be populated with list of addresses based on doctor and patient selected from first and second drop down (as shown below)


localhost:63342/AngularJS_Demos_Labs/Day2/day2_Labs/2_ng-options/index.html

Doctor:

Doctor2

Patient based on Doctor:

Doctor2_Patient2

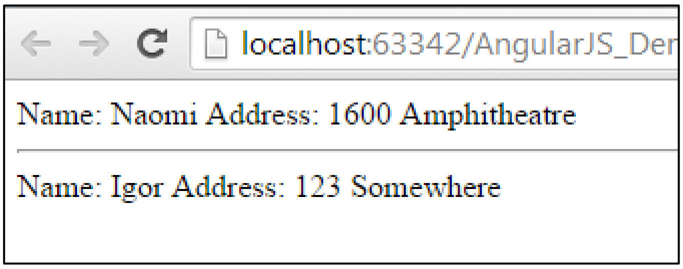
Address based on Patient:

Select Patient Addr

Select Patient Addr
Doctor2_Patient2_Addr_1
Doctor2_Patient2_Addr_2

[Lab2:2_drop_down_box]

- Lab on custom directive using isolated scope



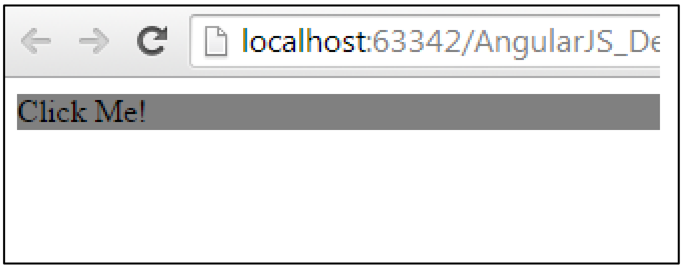
Name: Naomi Address: 1600 Amphitheatre

Name: Igor Address: 123 Somewhere

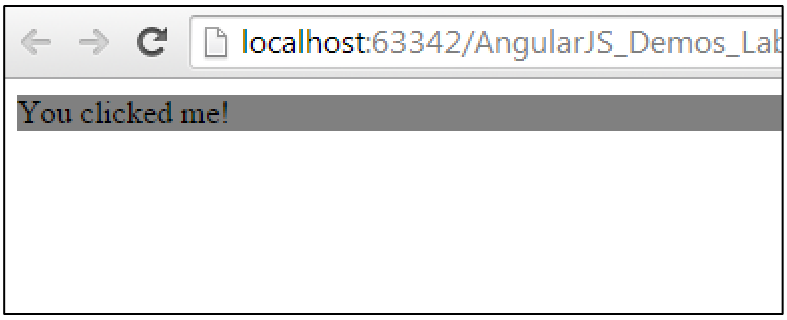
- Use custom directive <my-customer> two times in html page with its info attribute
- First time you use <my-customer> tag with info="Naomi"
- Second time you use <my-customer> tag with info="Igor"
- Correspondingly, as shown in image above, first tag should show you information (name & address) about customer whose name is "Naomi" and second tag should show you information about customer whose name is "Igor"
- In controller set the following to scope object:
- $scope.naomi = { name: 'Naomi', address: '1600 Amphitheatre'};
- $scope.igor = { name: 'Igor', address: '123 Somewhere' };
- Write the template url html file containing the html snippet to be displayed in place of tag

[Lab2:3_custom_directive_isolated_scope]

- Lab on custom directive using link function



Click Me!



You clicked me!

- Create a custom directive 'myDomDirective' with 'col' attribute for color (col = 'grey') and use it in view as follows:
- When you take mouse over it, its background-color should be same as the color value assigned to 'col' attribute. When you take mouse out of it its background-color should be white i.e. back to normal
- When you click on the custom directive, it should change its value from 'Click Me!' to 'You Clicked Me!'

  Lab2: 4 custom directive link func

  Lab2: 4 custom directive ex

- Similar example as above using fadein-fadeout feature (using jquery)
  Lab2: 4 custom direc fadein fadeout ex link

- Similar examples above using compile function in addition to link function.
  Lab2: 4 custom directive using link compile

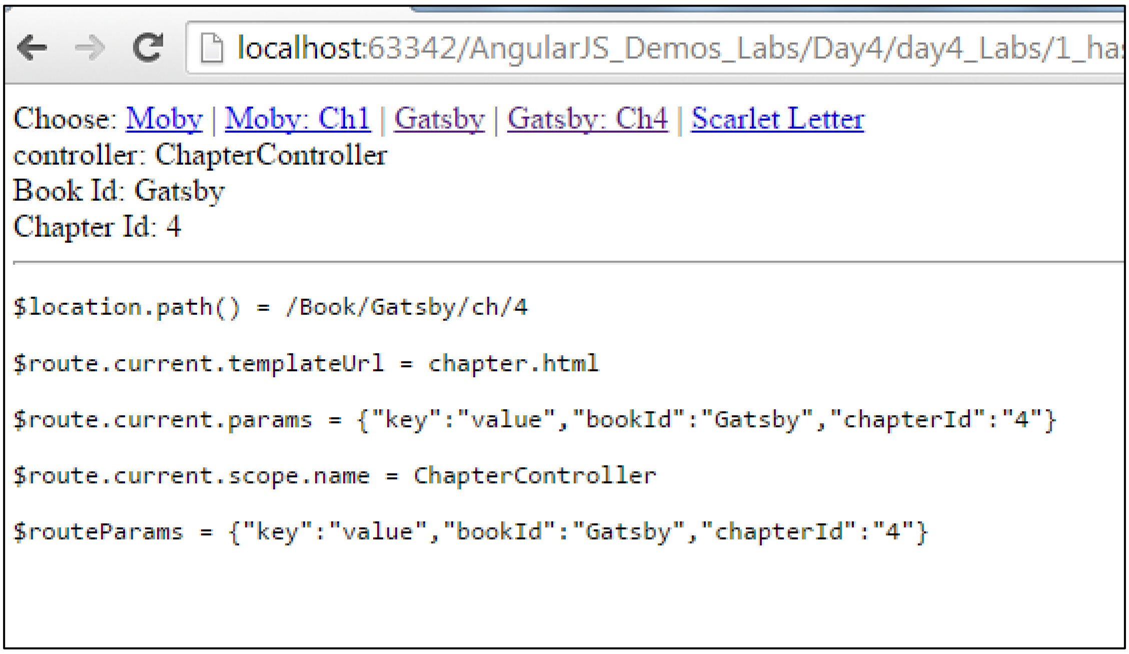
3. Lab 3

- Create/write a service function having functions to add, multiply, subtract and divide two numbers.

  Lab3: 1 service lab

4. Lab 4

- Create a lab on routing as follows:



localhost:63342/AngularJS_Demos_Labs/Day4/day4_Labs/1_has

Choose: Moby | Moby: Ch1 | Gatsby | Gatsby: Ch4 | Scarlet Letter
controller: ChapterController
Book Id: Gatsby
Chapter Id: 4

```
$location.path() = /Book/Gatsby/ch/4

$route.current.templateUrl = chapter.html

$route.current.params = {"key":"value","bookId":"Gatsby","chapterId":"4"}

$route.current.scope.name = ChapterController

$routeParams = {"key":"value","bookId":"Gatsby","chapterId":"4"}
```

- Provide following 2 routes for following hrefs:

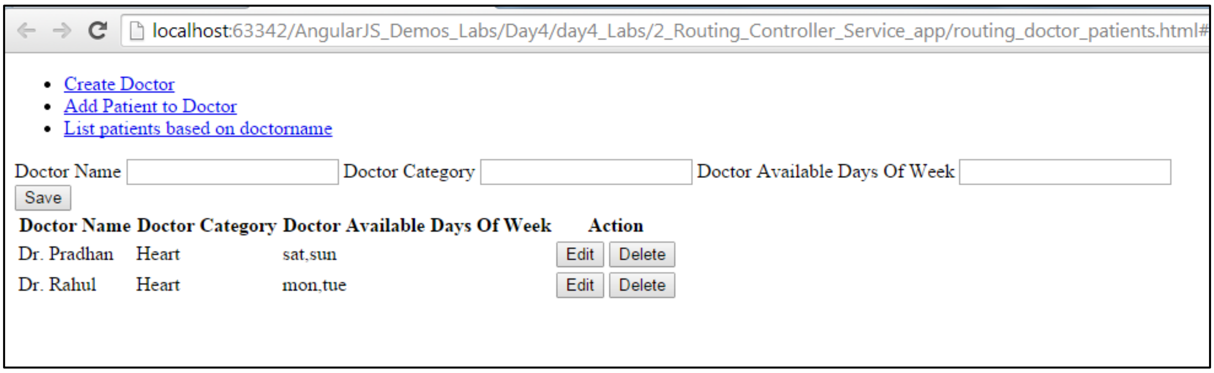
```
1st route) -    /Book/:bookId
And
2nd route) -    /Book/:bookId/ch/:chapterId

   <a href="Book/Moby">Moby</a>
   <a href="Book/Moby/ch/1">Moby: Ch1</a>
    <a href="Book/Gatsby">Gatsby</a>
    <a href="Book/Gatsby/ch/4?key=value">Gatsby: Ch4</a>

    <a href="Book/Scarlet">Scarlet Letter</a><br/>
```

- Write two controllers ‘book controller’ and ‘chapter controller’ to handle both routes.
Lab4: 1 hashless routing

- Create a mini SPA as follows:



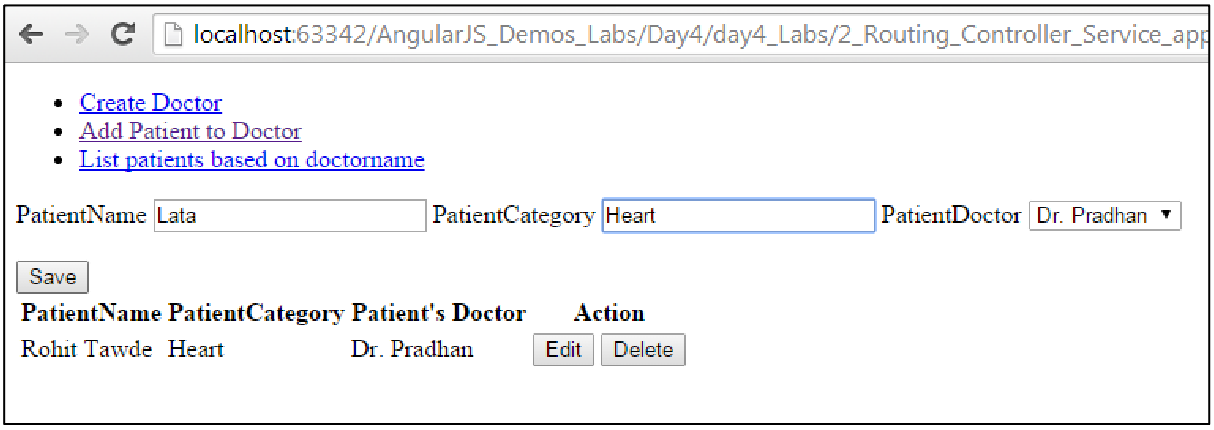
localhost:63342/AngularJS_Demos_Labs/Day4/day4_Labs/2_Routing_Controller_Service_app/routing_doctor_patients.html#

- Create Doctor
- Add Patient to Doctor
- List patients based on doctorname

Doctor Name [ ] Doctor Category [ ] Doctor Available Days Of Week [ ]
Save

Doctor Name	Doctor Category	Doctor Available Days Of Week	Action
Dr. Pradhan	Heart	sat,sun	Edit Delete
Dr. Rahul	Heart	mon,tue	Edit Delete

- Create Doctor – It should allow you to add doctor as shown above
- Add Patient to Doctor – It should allow you to add patient details to existing doctors



localhost:63342/AngularJS_Demos_Labs/Day4/day4_Labs/2_Routing_Controller_Service_app

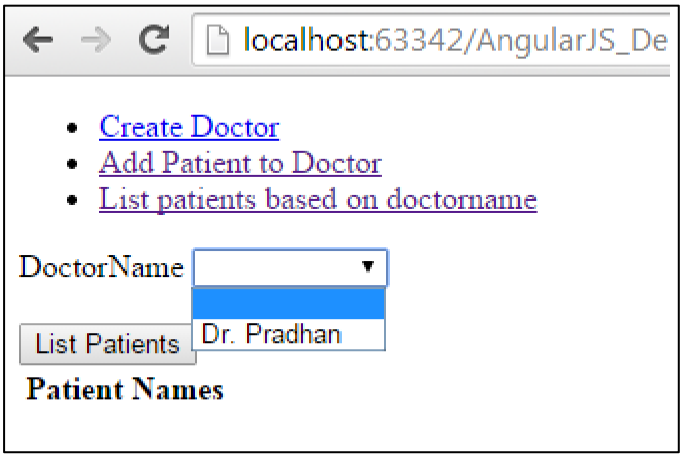
- Create Doctor
- Add Patient to Doctor
- List patients based on doctorname

PatientName Lata PatientCategory Heart PatientDoctor Dr. Pradhan ▾

Save

PatientName	PatientCategory	Patient's Doctor	Action
Rohit Tawde	Heart	Dr. Pradhan	Edit Delete

- List Patients based on doctorname – This should list patients when you select the Doctor's name from drop down.



localhost:63342/AngularJS_De

- Create Doctor
- Add Patient to Doctor
- List patients based on doctorname

DoctorName ▾

Dr. Pradhan

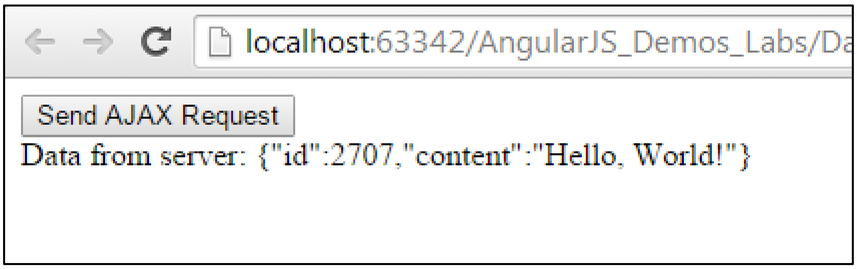
List Patients

Patient Names

Lab4: 2 routing doctor patients

5. Lab 5

- Send Ajax request using angular implicit promise. Write success and failure callback functions for implicit promise returned by $http.get() function.



localhost:63342/AngularJS_Demos_Labs/Da

Send AJAX Request

```
Data from server: {"id":2707,"content":"Hello, World!"}
```

Lab5:1 http get using promise

- Repeat above example using $q.defer() using explicit promise. Write code to send async Ajax call in factory function and explicitly return promise object to controller. In controller write the 'then' success callback function.

  Lab5:2_http_get_using_q_defer

Note: For any doubt/solution of these todo labs contact namrata.marathe@citiustech.com