

Internet Programming

Practice #10

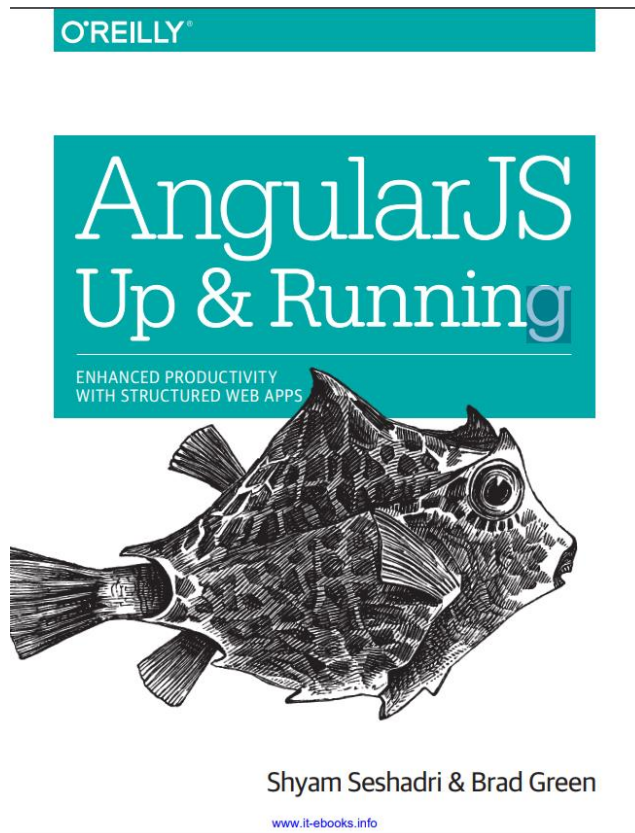
AngularJS - Lab 1



Dept. of Software and Information
Systems Engineering

By Shir Frumeramn based on Hasidi Netanel slides

Excellent book



Available to you on moodle

https://moodle1370/php.elifnigulp/el doom/li.ca.ugb.2%20SJralugnA/2/tnetnoc/ecruoser_dom/620Up%20And%20Running.pdf

Why AngularJS?

Official Documentation

“AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you currently have to write. And it all happens within the browser, making it an ideal partner with any server technology...”



Why AngularJS?

Features

- AngularJS is perfect for Single Page Applications (SPAs)
- AngularJS provides developers options to write client side application (using JavaScript) in a clean MVC (Model View Controller) way.
- Application written in AngularJS is cross-browser compliant. AngularJS automatically handles JavaScript code suitable for each browser.
- AngularJS is open source, completely free, and used by thousands of developers around the world

What is AngularJS?

- It is a client-side JavaScript framework
- It extends HTML DOM with additional attributes and makes it more responsive to user actions
- Our HTML code runs each time different JavaScript code (view) results in dynamic web page
- It works in a concept of MVC (Model-View-Controller) (Actually implement more like MVVM)

MVC in AngularJS

- **Model:**

- The data that we work with:

Can be a primitive type such as string, number, boolean or a complex type such as object

- **View:**

- Display content and data to a user in a browser
ng-app, ng-view

- **Controller:**

- Preforms the functionality and the interaction between the view and the model - \$scope
- Different controllers for different areas in our application

How to use it?

- AngularJS is distributed as a JavaScript file, and can be added to a web page with a script tag:

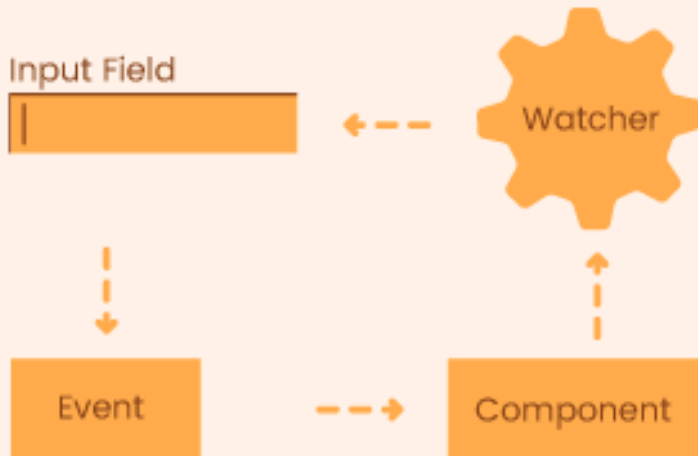
```
<script  
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js">  
</script>
```

Data Binding

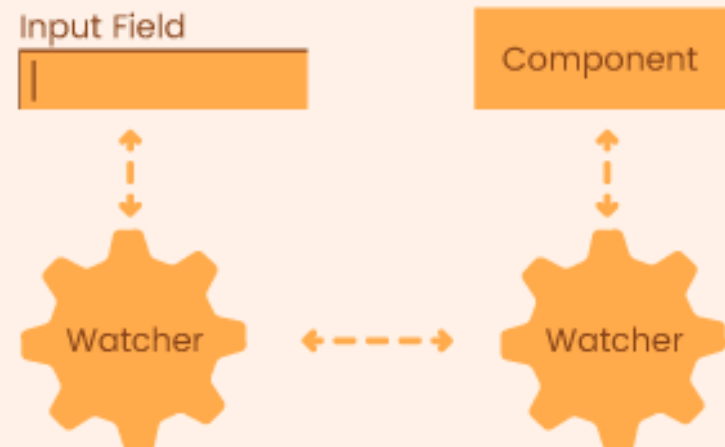
- **One way** - we take data coming from the server (or any other source), and update the Document Object Model (DOM)
- **two-way** - data-binding ensures that our controller and the UI share the same model, so that updates to one (either from the UI or in our code) update the other automatically

Data Binding

1 way data binding

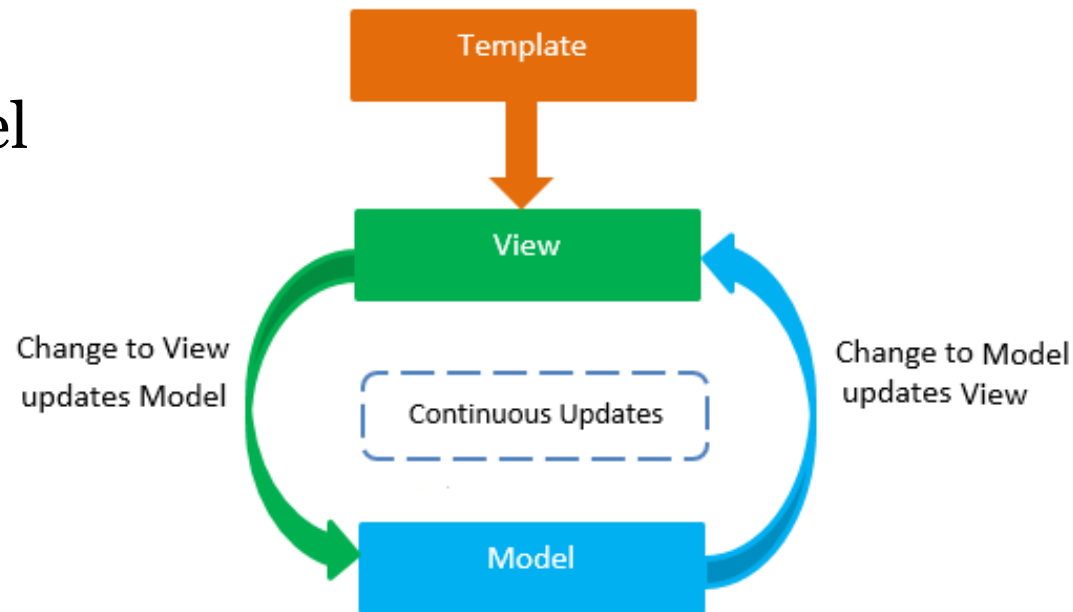


2 way data binding



2 way Data Binding

- Allows automatic synchronization of the data between the model and the view
- When model changes, the view reflects the change, and vice versa (two way binding).



Two-Way Data Binding

CodesJava.com

Data Binding Without Angular

- We need to use getters and setters from and to DOM elements.
- We need to handle the changing events.
- This is for just one field !

```
<script type="text/javascript">
```

```
function changeLabel() {  
    var textBox = document.getElementById("txb");  
    var label = document.getElementById("lbl");  
    label.innerHTML = textBox.value;
```

```
}  
setInterval(changeLabel, 500);
```

```
//or
```

```
//document.getElementById("txb").addEventListener("change", changeLabel);  
</script>
```

Data Binding With Angular

- We use the 'ng-model' to create model on the view that allows binding to the model (data).

```
<input type="text" ng-model="name" placeholder="Type something" />
```

- We use expressions to evaluate the model on the view

```
<label>{{ name }}</label>
```

Directives

- It's a marker on the HTML tag and tells Angular to run or reference some JavaScript code
- AngularJS:
 - lets you extend HTML with new attributes called **Directives**
 - has a set of built-in directives which offers functionality
 - AngularJS also lets you define your own directives
 - directives are extended HTML attributes with the prefix **ng-**

The most basic

The AngularJS framework can be divided into following three major parts –

- **ng-app** – directive that initializes an AngularJS application
- **ng-model** – directive that binds the value of HTML controls (input, select, textarea) to application data
- **ng-bind** – directive that initializes application data

Directives - Example

```
<!DOCTYPE html>
```

AngularJS starts automatically when the web page has loaded

```
<html>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
```

```
<body>
```

The **ng-app** directive tells AngularJS that the `<div>` element is the "owner" of an AngularJS **application** - only allowed to work on, control, and modify that particular section of the HTML

```
<div ng-app="">
```

```
<p>Name:
```

```
<input type="text" ng-model="name">
```

The **ng-model** directive binds the value of the input field to the application variable **name**

```
</p>
```

```
<p ng-bind="name"></p>
```

The **ng-bind** directive binds the innerHTML of the `<p>` element to the application variable **name**.

```
</div>
```

```
</body>
```

```
</html>
```

Expressions

- AngularJS binds data to HTML using **Expressions**
- can be written inside double braces: `{{ expression }}`
- AngularJS expressions can also be written inside a directive: `ng-bind="expression"`
- AngularJS will resolve the expression, and return the result exactly where the expression is written

Expressions - Example

Example: Let AngularJS change the value of CSS properties.

Change the color of the input box below, by changing its value:

lightblue

```
<div ng-app="" ng-init="myCol='lightblue'">  
  
<input style="background-color:{{myCol}}" ng-model="myCol">  
  
</div>
```

Expressions - Example

- Objects:

```
<div ng-app="" ng-init="person={firstName:'John',lastName:'Doe'}">  
  <p>The name is {{ person.lastName }}</p>  
  <p>The name is <span ng-bind="person.lastName"></span></p>  
</div>
```

- Arrays:

```
<div ng-app="" ng-init="points=[1,15,19,2,40]">  
  <p>The third result is {{ points[2] }}</p>  
</div>
```

- For More examples:

https://www.w3schools.com/angular/angular_expressions.asp

Module

- An AngularJS module defines an application
- Modules are AngularJS's way of packaging relevant code under a single name (package)
- The module is a container for the application controllers, services etc.
- The module can also depend on other modules as dependencies ([])

Module

- A module is created by using the AngularJS function **angular.module**

```
var app = angular.module("myApp", []);
```

- The first argument is the *name* of the module
- The second argument is an *array* of module names that this *module depends on*

Now you can add controllers, services and more, to your AngularJS application

Module

Creating a module

```
let app = angular.module('myApp', []);
```

app.js file

module as container

```
app.controller("myCtrl", [function(){
```

```
...
```

```
}]})
```

myCtrl.js file

```
<!DOCTYPE html>
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/
angularjs/1.6.9/angular.min.js"></script>

<script src="myCtrl.js"></script>
<script src="app.js"></script>

<body>

<div ng-app="myApp" ng-controller="myCtrl">
</div>

</body>

</html>
```

HTML file

Controller

- Fetching the right data from the server for the current UI
- Deciding which parts of that data to show to the user
Presentation logic, such as how to display elements, which parts of the UI to show, how to style them, etc.
- User interactions, such as what happens when a user clicks something or how a text input should be validated
- Gateway between the Model (data) and the View (UI)
- We use it through the ‘**ng-controller**’ directive.

Controller

- Creating the controller is done by attaching it to our module

```
angular.module('myApp')  
  .controller('MainCtrl', [function() {  
    // Controller-specific code goes here  
    console.log('MainCtrl has been created');  
  }]);
```

- Using through the 'ng-controller' directive.

```
<div ng-app="myApp">  
<head> <title>Notes App</title> </head>  
<body ng-controller="MainCtrl as ctrl">  
  // View-specific code goes here  
</body>  
</div>
```

Controller



- We will use the *: controller as syntax*, and a proxy name (*self*) for *this* keyword

index.html

```
<body ng-controller="MainCtrl as ctrl">
  {{ctrl.helloMsg}} AngularJS.
```

MainCtrl.js

```
app.controller('MainCtrl', [function() {
  let self = this
  self.helloMsg = 'Hello '
  ...
}])
```


Controller - Example

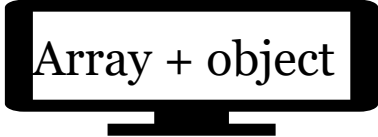


- The controller we wrote has no direct access to the view or any of the DOM elements that it needs to update. It is pure JavaScript. Creating Our First Controller
- When the user clicked the button and `changeMessage` was triggered, we did not have to tell the UI to update. It happened automatically.
- The HTML connects parts of the DOM to controllers, functions, and variables, and not the other way around.
- our whole aim in an AngularJS application should be to manipulate and modify the model (pure JavaScript), and let AngularJS do the heavy lifting of updating the UI accordingly

Built in Directives

- **ng-repeat**
 - one of the most versatile and heavily used directives of AngularJS, because it allows us to iterate over an array or over the keys and values of an object and display them in the HTML
 - the contents of the element on which the directive is applied is considered the template of the ng-repeat.
 - AngularJS picks up this template, makes a copy of it, and then applies it for each instance of the ng-repeat

Built in Directives



Array + object

- Notes on ng-repeat
 - For arrays use: `ng-repeat="note in ctrl.notes"`
 - For objects use: `ng-repeat="(key, value) in ctrl.notes"`
 - When we use the ng-repeat directive over an object, the keys of the object will be sorted in a case-sensitive, alphabetic order (Upper-case first)

Built in Directives

- Notes on ng-repeat

```
<div ng-repeat="note in ctrl.notes">
  <div>First Element: {{$first}}</div>
  <div>Middle Element: {{$middle}}</div>
  <div>Last Element: {{$last}}</div>
  <div>Index of Element: {{$index}}</div>
  <div>At Even Position: {{$even}}</div>
  <div>At Odd Position: {{$odd}}</div>
```

- `$first`, `$middle`, and `$last` are Boolean values that tell us whether that particular element is the first, between the first and last, or the last element in the array or object.
- `$index` gives us the index or position of the item in the array.
- `$odd` and `$even` tell us if the item is in an index that is odd or even (we could use this for conditional styling of elements, or other conditions we might have in our application).

Lab Exercise



- הוסיפו לקובץ הcontroller אובייקט של ערים, החזיקו עבור כל עיר: שם, מדינה, לינק לתמונה
- חברו את הcontroller לview, והציגו באמצעות ng-repeat אלמנט של פסקה עבור כל עיר, כך שבתוך הפסקה יופיעו פרטי העיר והתמונה

Any Questions?



And I'll come shortly..