



Ground Up

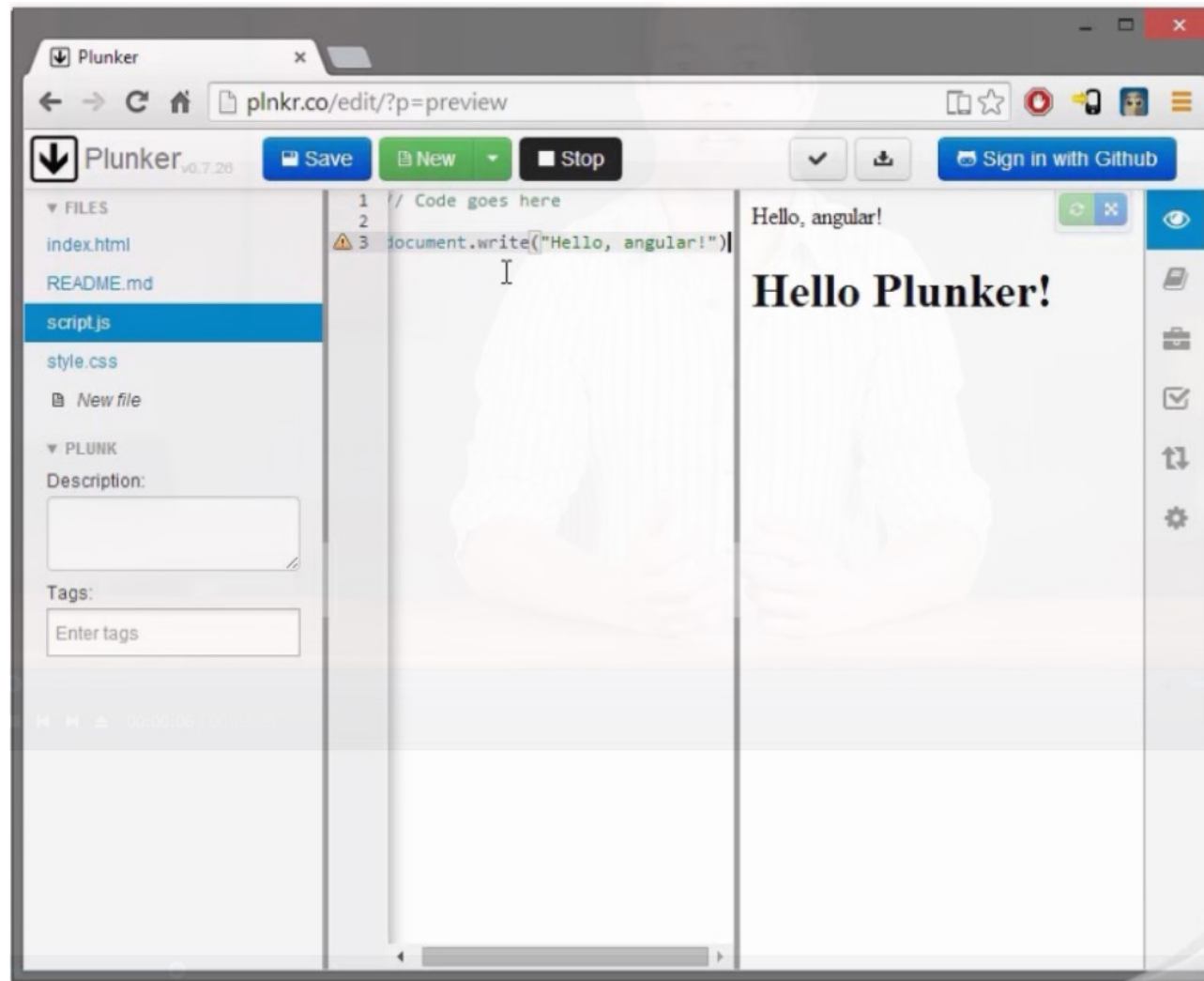
Syed Awase



3. ANGULARJS COMPONENTS



Plunker





First Angularjs Application

```
<script src="angular.js"></script>
```

- Two requirements
 - Add a **<script>** tag pointing to angular.js
- Add an ng-app attribute in your HTML
 - **ng-app** is an Angular **directive**
 - The **ng** is short for Angular

Example 3.1-basicapp

{{ one way read-access or display }}

```
<div ng-app>  
  This area controlled by Angular!  
</div>
```

Initializes an angular app



JavaScript Patterns

- Functions as abstractions
- Functions to build modules
- Functions to avoid global variables

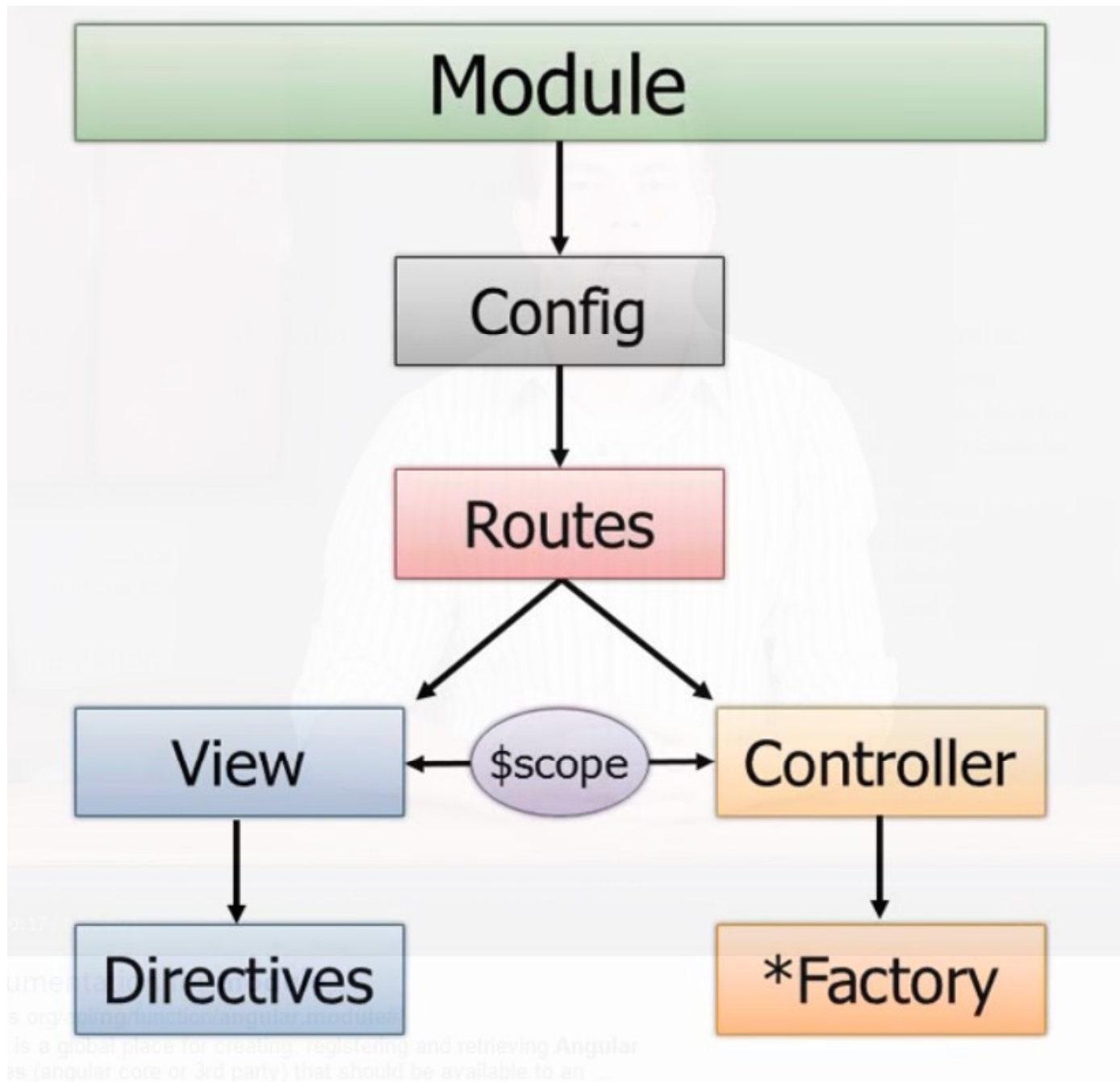
How JavaScript can be used a functional language



IIFE

- Immediately Invoke Function Expression
 - An anonymous function that invokes itself and is used to control the scope of the variables to build modules, to provide encapsulation and most of all avoid global variables.
 - IIFE Template

```
(function() {  
    // code goes here  
})();
```





Tracks all of the application code

- Loaded and template al template

Keeps the application modularized



Angular Module Method

Setter Method

```
angular.module("moduleName",[ "dependency"]);
```

Getter Method

```
angular.module("moduleName");
```



4 Main elements of AngularJS

Directives

- Fundamental building Block of angular
- serve as definitions for various declarations to use within an angular application

Binding Directive

Model Directive

Event Directive

Display Directive

Custom Directives

Factories

- Used to maintain one single of something
- Often used to maintain state.
- Additionally They can be used to provide an abstraction layer between your directives and your APIs

Controllers

- A lot of similar functionality to directive, but are predominantly used just for data connections and logical restraints around them. Unlike a directive, they can't contain any extra templating or complex ordering.

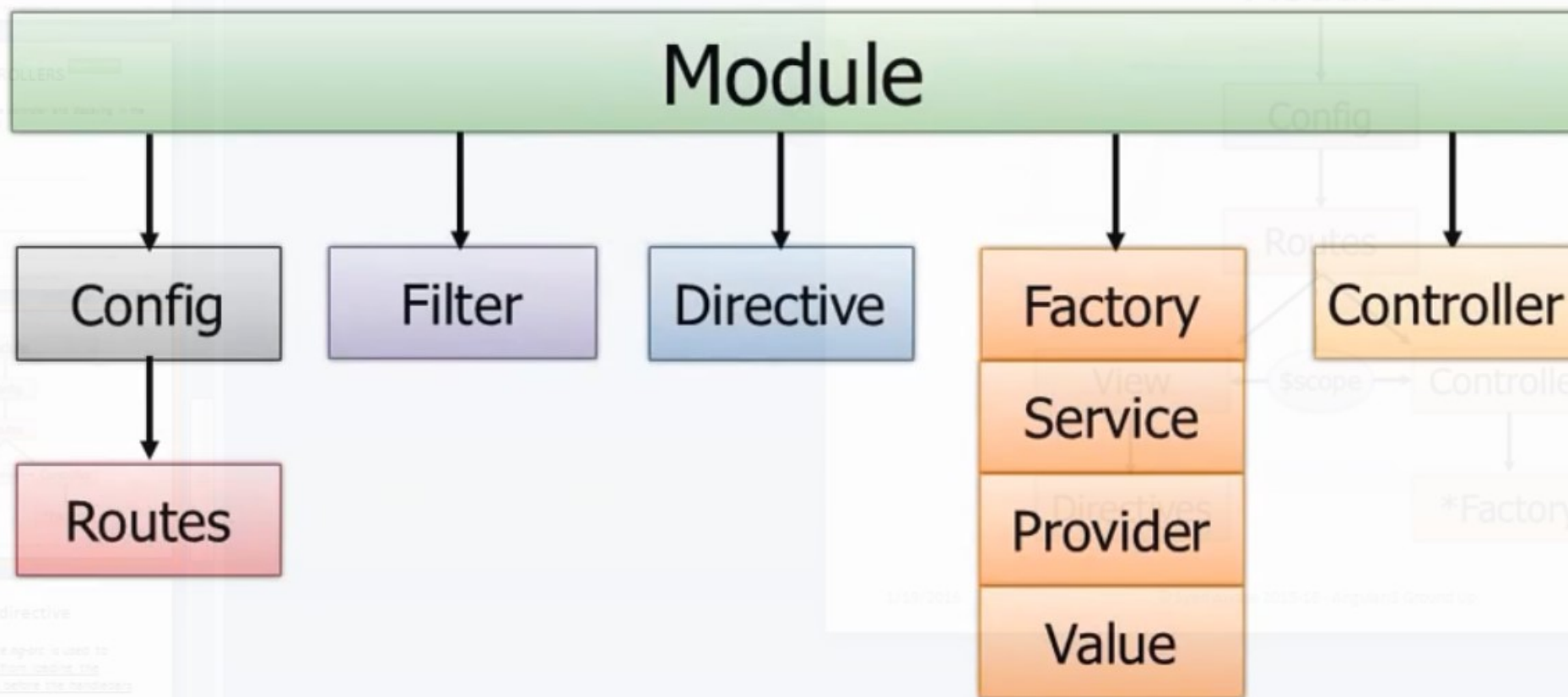
Filters

- Used to implement temporary transformations of data as it travels between the model and the view.



Modules are Containers

```
<html ng-app="moduleName">
```





CREATING A MODULE

The slide illustrates how to create an AngularJS module. It features two code snippets and a diagram.

Code Snippet 1:

```
<html ng-app="moduleName">
```

```
var demoApp = angular.module('demoApp', []);
```

Code Snippet 2:

```
var demoApp = angular.module('demoApp',  
  ['helperModule']);
```

Diagram:

A diagram titled "Modules are Containers" shows a central box labeled "Module". Below it, five boxes represent components that can be included in a module: "Filter", "Directive", "Factory", "Service", and "Controller". Arrows point from each of these component boxes up to the "Module" box, indicating that they are contained within it.

Annotations:

- A blue callout box points to the empty array `[]` in the first code snippet, asking: "What's the Array for?"
- A blue callout box points to the `'helperModule'` in the second code snippet, stating: "Module that demoApp depends on"



CONTROLLER

DEFINES THE MODEL

•**Products**

Implements actions with methods

```
angular.module("moduleName");
```



CONTROLLER

1

- Define Controller Function
 - Global Namespace
 - Or registered with a module

2

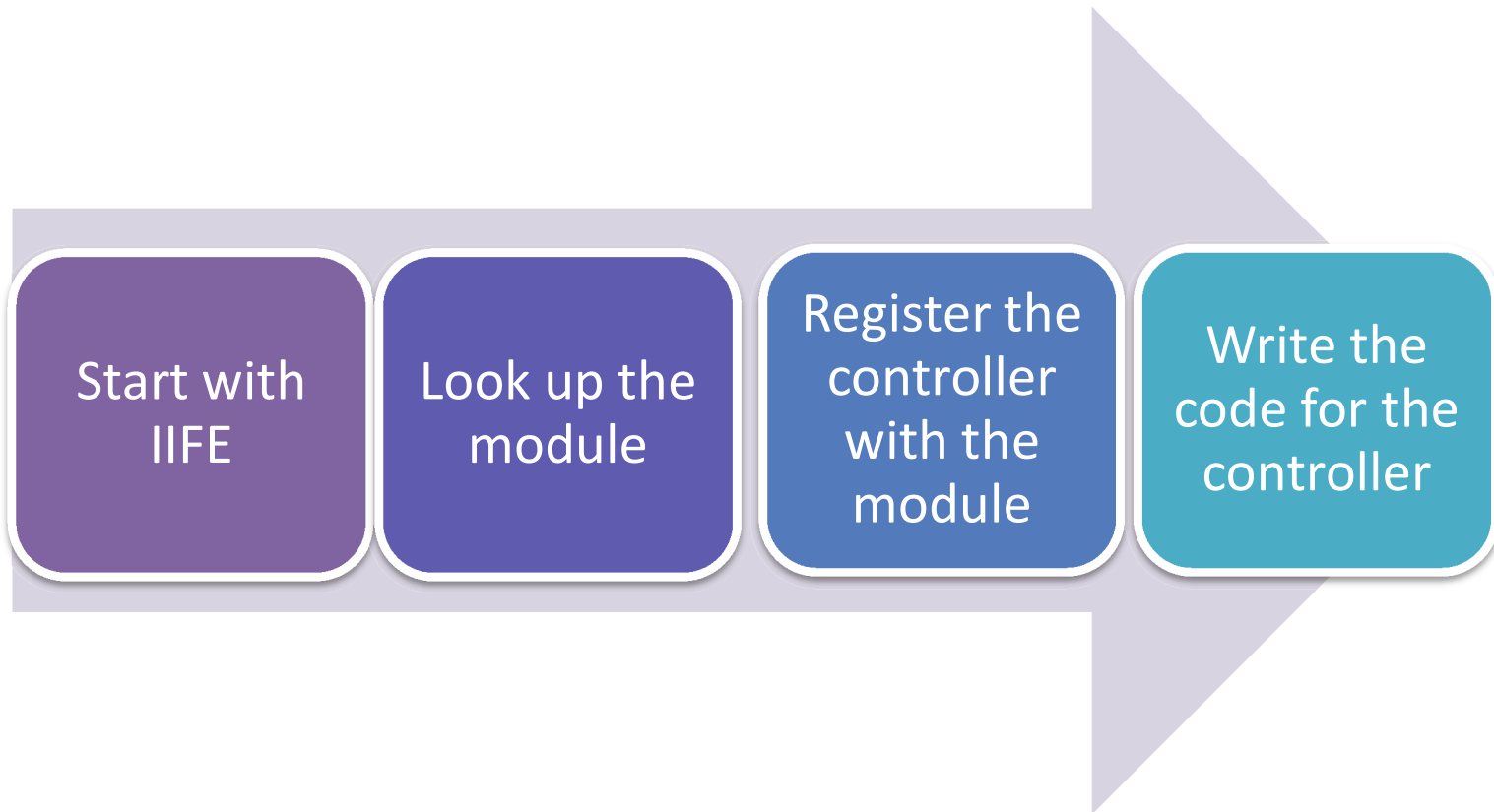
- Include ng-controller directive inside ng-app
- Controller function should be PascalCase

3

- A module can have multiple controllers.



Steps for Building a Controller





1.Start with IIFE

```
//IIFE Template

(function(){

    // Immediately Invoke Function Expression
    //Self Invoking function avoids global variable

})();
```

2.Look up the Module

```
angular
    .module("moduleName",[“dependencies”]);
```




3.Look up the Module

```
angular
    .controller("ControllerNameCtrl",["$scope", function($scope_{

    }));
```

4.Write the code for the controller

```
(function () {
    angular
        .module("productManagement")
        .controller("ProductListCtrl",
            ["$scope",
            function($scope) {
                // Code here
            }]);
})();
```



CREATING A CONTROLLER IN MODULE

```
var demoApp = angular.module('demoApp', []);

demoApp.controller('SimpleController', function ($scope) {
  $scope.customers = [
    { name: 'Dave Jones', city: 'Phoenix' },
    { name: 'Jamie Riley', city: 'Atlanta' },
    { name: 'Heedy Wahlin', city: 'Chandler' },
    { name: 'Thomas Winter', city: 'Seattle' }
  ];
});
```

Define a Controller

Define a Module



CONTROLLERS

- Central pieces of angular framework and are incharge of an area of an application.
- Controllers are incharge for building a model. A model contains the data we need to work with.
- 4 BASIC Facts of a Controller
 - Controller directive in HTML (ng-controller)
 - Controller will be a function that Angular invokes
 - Controller takes a ***\$scope*** parameter
 - ***Attach model to \$scope***

Example 3.3ctrlbasics



CONTROLLERS

– Capabilities

- Set up the model onto the scope object
- No direct manipulation of the HTML
- Separation of concerns – controller is focused on scope and html on views.
- Better factored
- Multiple Controllers can be used
- Complex Object
- Nest Controllers

Example 3.3ctrlbasics



View, Controllers and Scope



\$scope is the "glue" (ViewModel) between a controller and a view



\$Scope

- A built-in service communication mechanism between the view and the controller
- Angular injects \$scope into the controller
- Controller augments the scope



“Controller As”

- Angular 1.2 simplified \$scope handling by binding the model and the methods directly into the controller using “Controller As” syntax
- \$scope is not required as a parameter
- Model and methods are defined on the controller itself
- In the view, the model and the methods are referenced using an alias defined in the ng-controller i.e vm (viewmodel)
- \$scope still exists, but it is behind the scenes



ng-controller

CONTROLLERS

```
<div class="container" data-ng-controller="SimpleController">
  <h3>Adding a Simple Controller</h3>
  <ul>
    <li data-ng-repeat="cust in customers">
      {{ cust.name }} - {{ cust.city }}
    </li>
  </ul>
</div>

<script>
  function SimpleController($scope) {

    $scope.customers = [
      { name: 'Dave Jones', city: 'Phoenix' },
      { name: 'Jamie Riley', city: 'Atlanta' },
      { name: 'Heedy Wahlin', city: 'Chandler' },
      { name: 'Thomas Winter', city: 'Seattle' }
    ];

  }
</script>
```

Define the
controller to use

\$scope injected
dynamically

Access \$scope



CONTROLLERS

– Multiple Controllers

```
<body ng-app>
  <h1>Hello Angular!</h1>
  <!--date controller code goes here -->
  <div ng-controller="DateController">
    Current time is:{{date}}
  </div>
  <!--Quote controller code goes here -->
  <div ng-controller="QuoteController">
    <div>Stock:{{stock.symbol}}</div>
    <div>Price:{{stock.price}}</div>
  </div>
  <!--Chart controller code goes here -->
  <div ng-controller="ChartController">
    
  </div>
</body>
```



CONTROLLERS

- Attaching an object to the controller and displaying in the view. **Example :3.4ctrlex2**

```
var app = angular.module('plunker', []);

app.controller('MainCtrl', function($scope) {

    //create a person object
    var person = {
        firstName: "James",
        lastName: "Bond",
        imageSrc: "http://georgesjournal.files.wordpress.com/2012/02/007_at_50_ge_pierece_brosnan.jpg"
    };

    $scope.name = 'Angular';
    $scope.message = 'This is my first example using controllers';

    // attach the person object to the scope
    // try removing this and see that person object details do not appear
    $scope.person = person;

});
```

```
<div>
  <h1>Person Details </h1>
  <div> first name: {{person.firstName}}</div>
  <div> first name: {{person.lastName}}</div>
  <div> </div>
</div>
```



ng-src directive

- the angularjs directive ***ng-src*** is used to prevent the browser from loading the resource (e.g. image) before the handlebars get parsed.

```
<h1>Person Details </h1>
<div> first name: {{person.firstName}}</div>
<div> first name: {{person.lastName}}</div>
// we have replaced src with ng-src for angular js to parse it and load the image
<div> </div>
</div>
```

Example:3.5ctrllex3-ngsrc



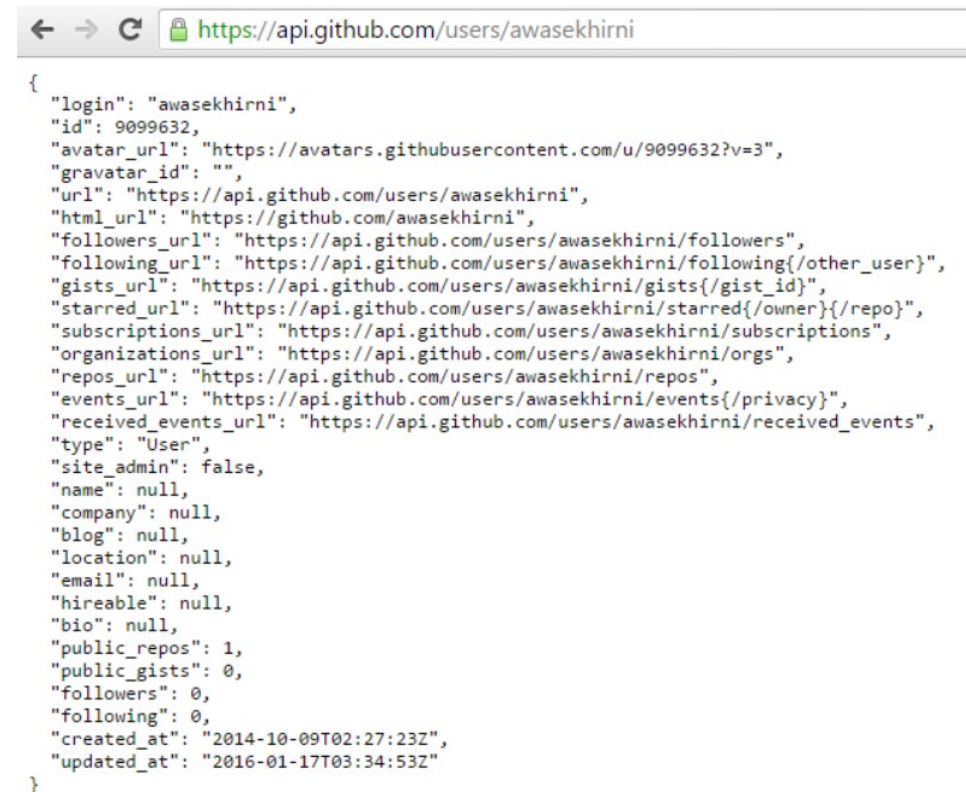
\$http Service

- Fetching/consuming data from web service using **\$http** Service (built-in Service)
- Encapsulated HTTP Communication service which provides an object with methods
 - GET, POST , PUT, DELETE
- Can “ask” for \$http inside a controller
 - Just add as another parameter to controller function
 - Asynchronous calls are made, and always return a promise
 - A promise to deliver a value in the future



GitHub API

- Available from JavaScript in a browser
- Return JSON(easy to convert into Objects)
 - Angularjs \$http services converts it into an object.
- No authentication or client required



The screenshot shows a web browser with the address bar displaying `https://api.github.com/users/awasekhirni`. The page content is a JSON object representing the user's profile data.

```
{
  "login": "awasekhirni",
  "id": 9099632,
  "avatar_url": "https://avatars.githubusercontent.com/u/9099632?v=3",
  "gravatar_id": "",
  "url": "https://api.github.com/users/awasekhirni",
  "html_url": "https://github.com/awasekhirni",
  "followers_url": "https://api.github.com/users/awasekhirni/followers",
  "following_url": "https://api.github.com/users/awasekhirni/following{/other_user}",
  "gists_url": "https://api.github.com/users/awasekhirni/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/awasekhirni/starred{/owner}/{repo}",
  "subscriptions_url": "https://api.github.com/users/awasekhirni/subscriptions",
  "organizations_url": "https://api.github.com/users/awasekhirni/orgs",
  "repos_url": "https://api.github.com/users/awasekhirni/repos",
  "events_url": "https://api.github.com/users/awasekhirni/events{/privacy}",
  "received_events_url": "https://api.github.com/users/awasekhirni/received_events",
  "type": "User",
  "site_admin": false,
  "name": null,
  "company": null,
  "blog": null,
  "location": null,
  "email": null,
  "hireable": null,
  "bio": null,
  "public_repos": 1,
  "public_gists": 0,
  "followers": 0,
  "following": 0,
  "created_at": "2014-10-09T02:27:23Z",
  "updated_at": "2016-01-17T03:34:53Z"
}
```

Example: `3.6ctrl-$httpService`



Open Movie Database API

- A free web service to obtain movie information, all content and images on the sites are contributed and maintained by users.

```
{
  "Title": "Batman",
  "Year": "1989",
  "Rated": "PG-13",
  "Released": "23 Jun 1989",
  "Runtime": "126 min",
  "Genre": "Action, Adventure",
  "Director": "Tim Burton",
  "Writer": "Bob Kane (Batman characters), Sam Hamm (story), Sam Hamm (screenplay), Warren Skaaren (screenplay)",
  "Actors": "Michael Keaton, Jack Nicholson, Kim Basinger, Robert Wuhl",
  "Plot": "Gotham City: dark, dangerous, 'protected' only by a mostly corrupt police department. Despite the best",
  "Language": "English, French",
  "Country": "USA, UK",
  "Awards": "Won 1 Oscar. Another 9 wins & 22 nominations.",
  "Poster": "http://ia.media-imdb.com/images/M/MV5BMTYwNjAyODIyMF5BM15BanBnXkFtZTYwNDMwMDk2._V1_SX300.jpg",
  "Metascore": "66",
  "imdbRating": "7.6",
  "imdbVotes": "253,443",
  "imdbID": "tt0096895",
  "Type": "movie",
  "Response": "True"
}
```

- <http://www.omdbapi.com/?t=Batman&y=&plot=full&r=json>

Example: `3.7ctrl-$http-omdb`



MODULES

- Controllers usually live in modules
 - Avoids the global namespace

We can have Multiple Modules for a specific application.
- Working with modules
 - **Create a module with a name**
 - *var app = angular.module("githubViewer",[]);*
 - **Register your controllers in the module**
 - *App.controller("MainController",MainController);*
 - **Tell Angular to use your module with ng-app**
 - *<body ng-app="githubViewer"></body>*



CONTROLLER BEST PRACTICES

- Define each controller as a separate javascript file to keep concerns separate.
- Name the controller in PascalCase.
- Suffix the controller name with “Controller” or “ctrl”
- Wrap the controller in an Immediately Invoked function expression (IIFE)



Summary

- 4 elements of angularjs
- Controllers manipulate scope
- Controllers can live in a module
- Controllers can use services like \$http
- Methods return promise object and we get the data using the .then method