ExcelShore® Operational Excellence

Cybage® Delivering Value Scientifically

Introduction to AngularJS

Authored and Presented by : Sushant Banerjee
Email : sushantba@cybage.com | Extn : 7221

Learning Objectives

- After attending the sessions on AngularJS, 70% participants will be able to understand how AngularJS can be implemented in the projects.
- After practicing AngularJS at least for 2 weeks and completing all assignments, the participants will be able implement Angular in the projects.

Course Structure

Target audience	Any developer who
Level	Beginner
Pre-requisites	HTML, JavaScript
Training methods	Lecture, Demonstration
Evaluation	Objective type test.

Cybage®
Delivering Value Scientifically

Agenda

- Introduction
- SPA
- MVC Pattern
- Module
- Controller
- Template
- Data Binding
- Directives
- Forms
- Services
- Filter
- Routing
- Unit Test

Cybage®
Delivering Value Scientifically

What is AngularJS

- A project by Google
  - Developed and maintained by Google
- A JavaScript framework
  - Can be added to HTML file using script tag
- Helps you to build dynamic web applications
  - Allows you to extend HTML vocabulary
  - Extends HTML attributes with directives
- A complete client side solution
  - Presents higher level of abstraction
  - No need to write DOM manipulation and AJAX code.

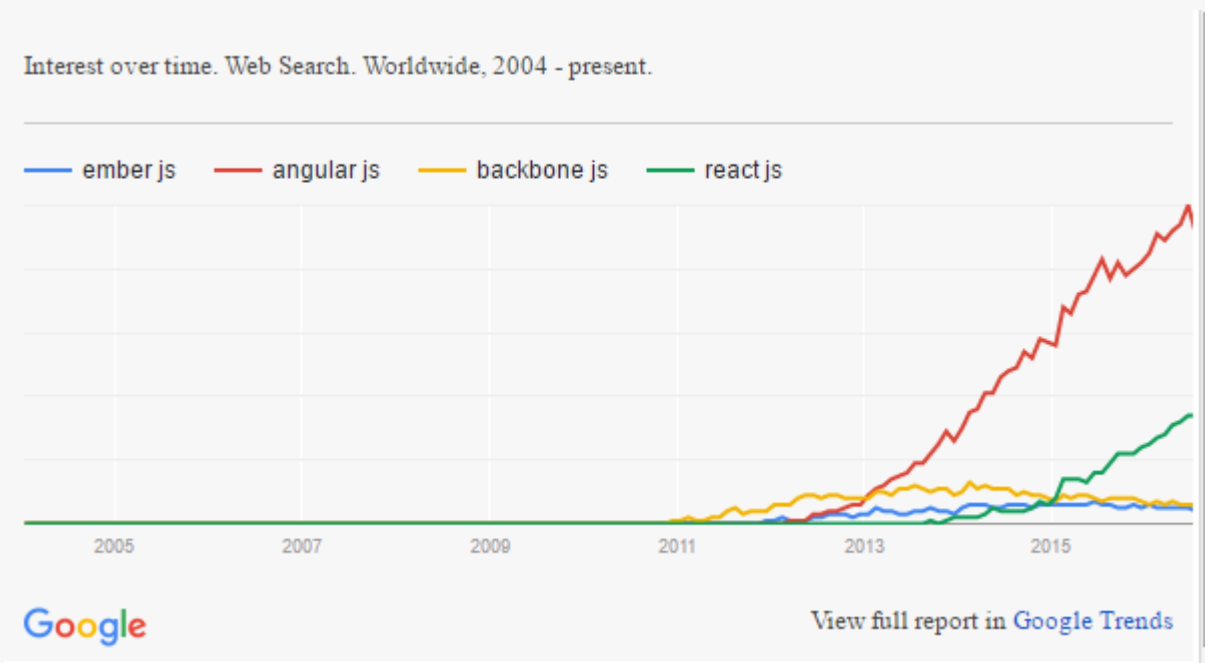
Cybage®
Delivering Value Scientifically

Why AngularJS

- Right structure for rapid development
- Long-term maintainability
- Modularity and reusability
- Testability and reliability
- Separation of concern.

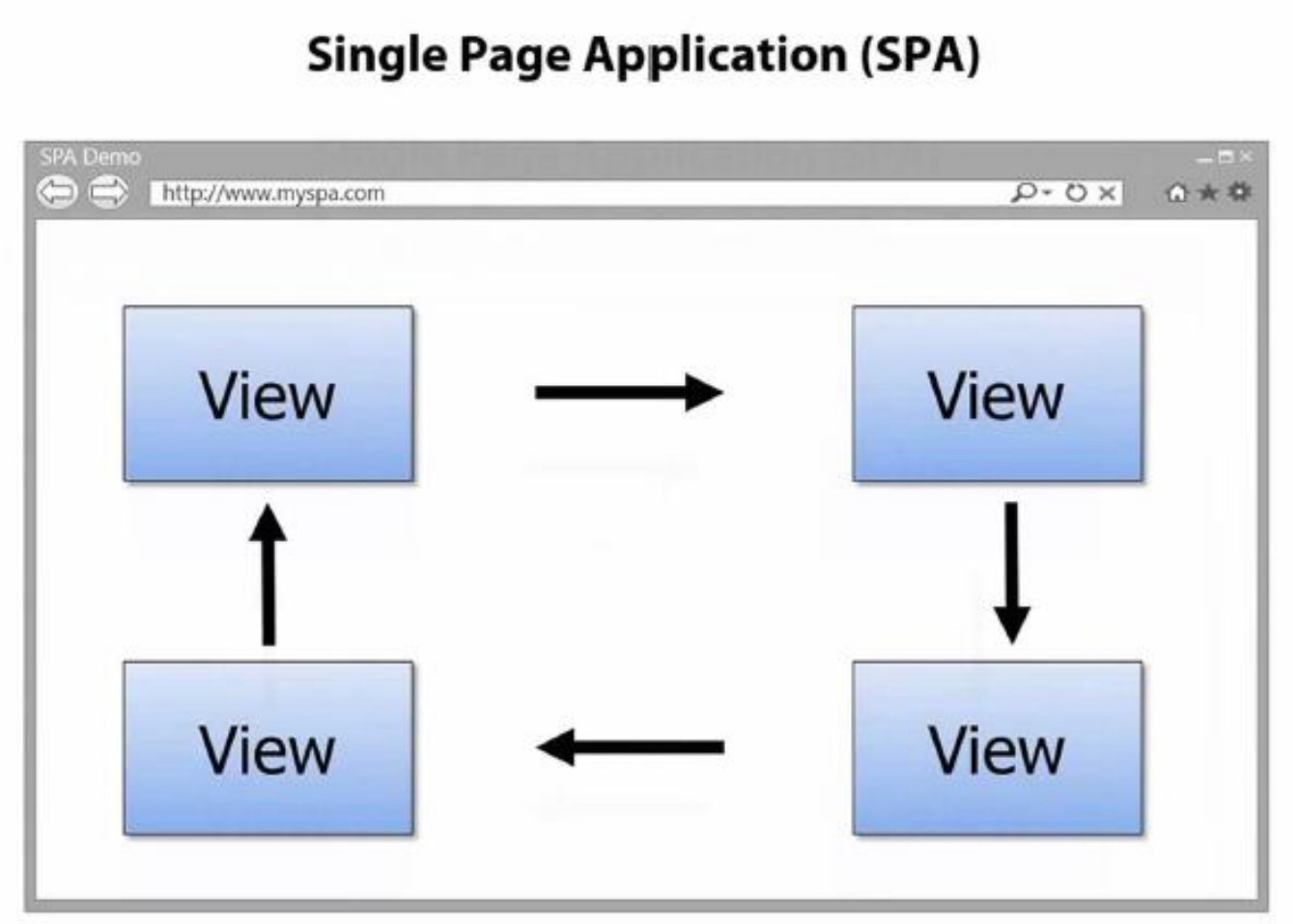
JavaScript MV* frameworks

- Ember is used by Yahoo!
- Angular powered by Google
- Backbone is used to realize WordPress.com and is a part time project of Jeremy Ashkenas, currently employed by the New York Times
- React is introduced and powered by Facebook.



Cybage®
Delivering Value Scientifically

What is SPA



Cybage®
Delivering Value Scientifically

The Problems

The Challenge with SPAs

DOM Manipulation

History

Module Loading

Routing

Caching

Object Modeling

Data Binding

Ajax/Promises

View Loading

Cybage®
Delivering Value Scientifically

The Solution

Data Binding | MVC | Routing | Testing

jqLite | Templates | History | Factories

AngularJS is a full-featured SPA framework

ViewModel | Controllers | Views | Directives

Services | Dependency Injection | Validation

Cybage®
Delivering Value Scientifically

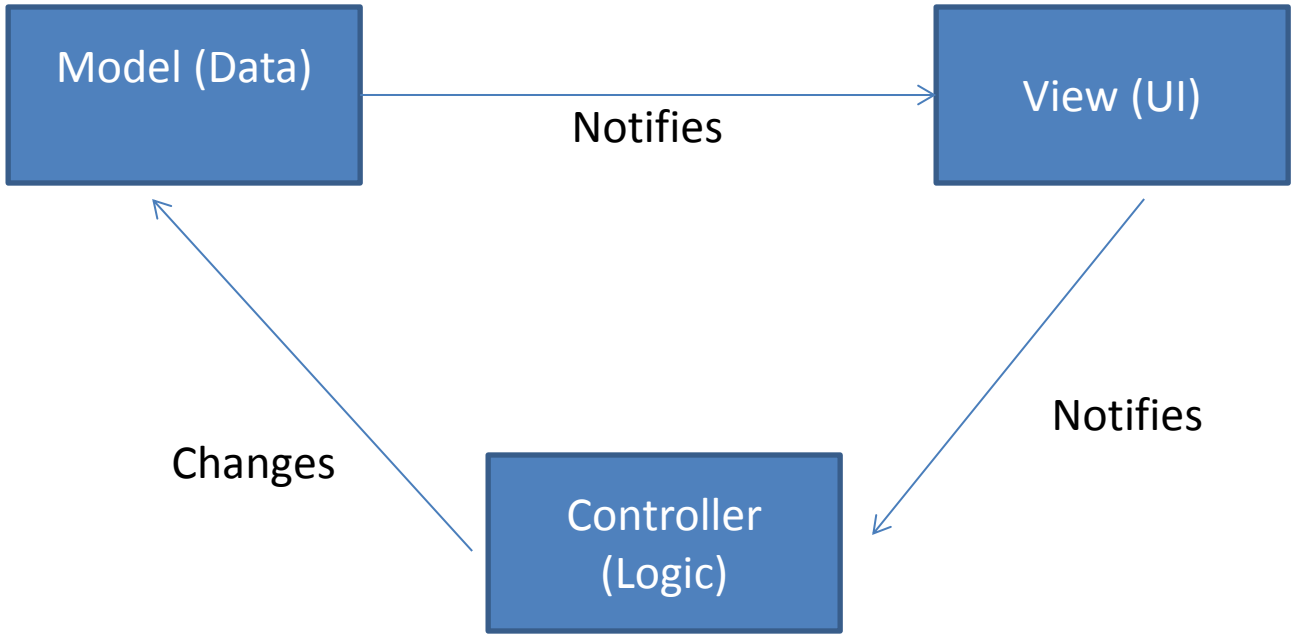
Get AngularJS

Starting AngularJS

- Loading AngularJS library
  - Using Google CDN or
  - Download and refer local version
- Bootstrapping AngularJS
  - Using ng-app
  - Denoting area controlled by Angular

Demo

MVC Pattern



Module

What is a module?

- A package with relevant code
- Can contain controllers, services, directives etc.
- A module can depend on other modules
- Angular uses module to bootstrap applications

Cybage®
Delivering Value Scientifically

Module

How to create a module?

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

- First parameter – name of a module
- Second parameter – array of dependent modules

Creating module with dependencies

```
angular.module('myApp', ['module1', 'module2']);
```

Loading an existing module

```
angular.module('myApp');
```

AngularJS bootstrap applications using module.

```
<html ng-app="myApp"></html>
```

Demo

Controllers

What is a controller?

- JavaScript functions called by Angular
- Works as a gateway between model and view
- Always linked to UI

Responsibilities of a controller

- Can fetch data from the server
- Decides what data to show
- Handles presentation logic
- Handles user input and validation.

Cybage®
Delivering Value Scientifically

Uses of Controllers

When to use controllers

- Adding state to $scope object
- Adding behavior to $scope object

When not to use controllers

- To manipulate DOM – instead use data binding & directives
- To format input – instead use form controls
- To filter output – instead use angular filters
- To share code/state across controllers – instead use angular services

ng-controller directive

- Angular instantiate a new controller
- By calling controller's constructor function
- A new child scope as $scope is created
- $scope is injected to controller's constructor function.

Demo

Cybage®
Delivering Value Scientifically

$scope Versus controllerAs Syntax

- In AngularJS 1.2 and later, a new syntax is introduced
- The controllerAs syntax
- It allows us to define the variables on the controller instance using the **"this"** keyword
- You can refer to them through the controller from the HTML.

Demo

Template

What is a template?

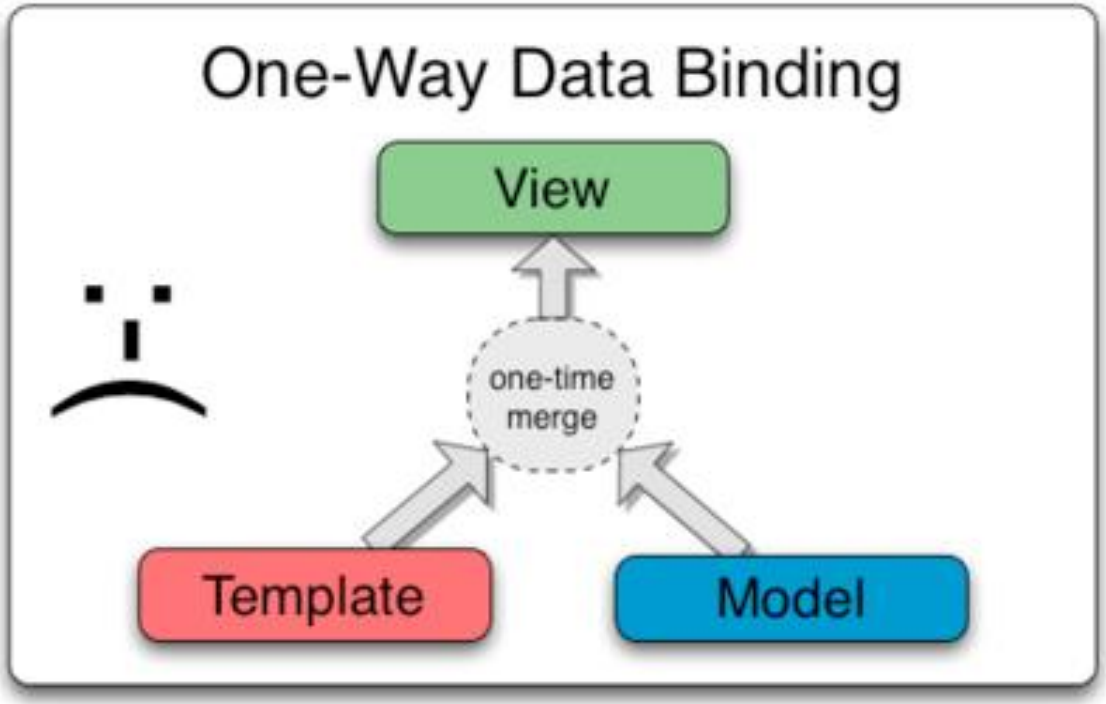
- HTML with Angular specific elements and attributes
- Creates dynamic view combining controllers and models

Angular elements and attributes used in templates

- Directives
  - Apply special behavior to HTML elements or attributes
- Markup
  - The double curly brace notation {{ }} that binds expression to elements
- Filter
  - Formats data for display
- Form Controls
  - Validates user input.

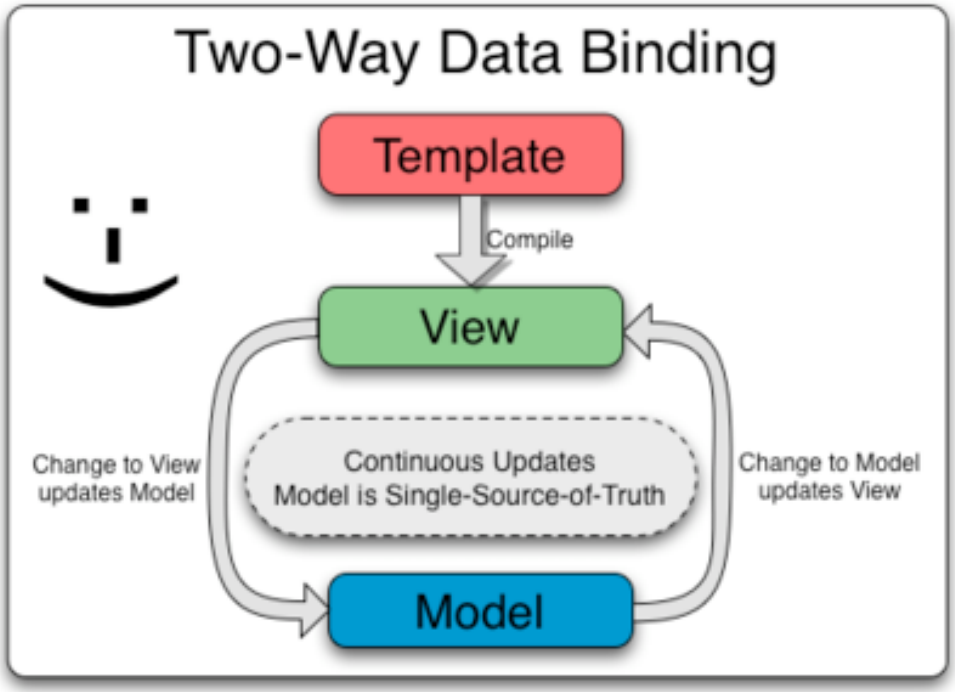
Data Binding – One Way

- Data Binding in Classical Template Systems



Data Binding – Two Way

- Data Binding in Angular Templates



Demo

Directives

Directives

What are directives?

- Markers on DOM element
- An attribute or element that manipulate DOM
- Tells Angular to attach specified behavior to DOM elements

Where to apply

- Html tags, attributes, class and comments

```
<my-dir></my-dir>
<span my-dir="exp"></span>
<!-- directive: my-dir exp -->
<span class="my-dir: exp;"></span>
```

Built-in directives

ng-app

- Automatically initializes Angular application

ng-controller

- Calls controller function and initializes

ng-model

- Stores or updates the value of input field into/from application variables

ng-bind

- Binds application variables to HTML elements

ng-click

- Evaluates any expression passed to it when the button is clicked

ng-init

- Used to initialize application variables.

Cybage®
Delivering Value Scientifically

ng-bind & Angular Expression

ng-bind

- Binds application variables to HTML elements
- If value changes UI updated automatically

Angular expression

- Double curly braces {{ }} used to write expression
- Can bind variables just like ng-bind and keep up to date
- There is no function difference between ng-bind and double curlies.

Built-in directives

ng-repeat

- It allows us to iterate over an array or over the keys and values of an object and display them in the HTML
- Basically the same as a for each loop in any programming language

Built-in directives

ng-show

- Shows HTML elements only when a variables value is truthy

ng-hide

- Hides HTML elements only when a variables value is truthy

What is truthy

- True
- Nonempty strings
- Nonzero numbers
- Nonnull JS objects

Demo

Forms

Data Binding in Forms

- ng-bind
  - Can read model data and show in UI
- Double curly braces {{ some expression }}
  - Can do the same as ng-bind
- ng-model
  - Can capture user input and update model data.

Submitting a Form

- ng-click
  - Submits a form when user clicks a button
  - Calls a function in the controller
- ng-submit
  - Submits a form when user clicks a button
  - Submits a form even when user press enter.

Form States in Angular

Form State	Description
$invalid	AngularJS sets this state when any of the validations (required, ng-minlength, and others) mark any of the fields within the form as invalid.
$valid	The inverse of the previous state, which states that all the validations in the form are currently evaluating to correct.
$pristine	All forms in AngularJS start with this state. This allows you to figure out if a user has started typing in and modifying any of the form elements. Possible usage: disabling the reset button if a form is pristine.
$dirty	The inverse of $pristine, which states that the user made some changes (he can revert it, but the $dirty bit is set).
$error	This field on the form houses all the individual fields and the errors on each form element.

Cybage®
Delivering Value Scientifically

Validators in Angular

Validator	Description
required	As previously discussed, this ensures that the field is required, and the field is marked invalid until it is filled out.
ng-required	Unlike required, which marks a field as always required, the ng-required directive allows us to conditionally mark an input field as required based on a Boolean condition in the controller.
ng-minlength	We can set the minimum length of the value in the input field with this directive.
ng-maxlength	We can set the maximum length of the value in the input field with this directive.
ng-pattern	The validity of an input field can be checked against the regular expression pattern specified as part of this directive.
type="email"	Text input with built-in email validation.
type="number"	Text input with number validation. Can also have additional attributes for min and max values of the number itself.
type="date"	If the browser supports it, shows an HTML datepicker. Otherwise, defaults to a text input. The ngmodel that this binds to will be a date object. This expects the date to be in yyyy-mm-dd format (e.g., 2009-10-24).
type="url"	Text input with URL validation.

Displaying Error Messages

- Using “name” attribute of HTML inputs
  - Creates a model on the form for the particular field along with error state
- $error state
  - We can use name attribute of inputs to check the state
  - And display error message depending on the state
- $invalid state
  - Shows error message unless all the validations are successful.

Highlighting Input Fields

- AngularJS adds and removes following CSS classes to and from the forms and input elements.

Form state	CSS class applied
$invalid	ng-invalid
$valid	ng-valid
$pristine	ng-pristine
$dirty	ng-dirty

Input state	CSS class applied
required	ng-valid-required or ng-invalid-required
min	ng-valid-min or ng-invalid-min
max	ng-valid-max or ng-invalid-max
minlength	ng-valid-minlength or ng-invalid-minlength
maxlength	ng-valid-maxlength or ng-invalid-maxlength
pattern	ng-valid-pattern or ng-invalid-pattern
url	ng-valid-url or ng-invalid-url
email	ng-valid-email or ng-invalid-email
date	ng-valid-date or ng-invalid-date
number	ng-valid-number or ng-invalid-number

Demo

Services in AngularJS

What are Angular Services

- Angular services are functions or objects
- Can hold behavior or state across application
- Angular service instantiated only once
- AngularJS service can have following functionalities
  - Repeated behavior
  - Shared state
  - Caches
  - Factories.

Problems with Controllers

- Controller instances created and destroyed as we navigate across the application
- One controller cannot directly communicate with another controller to share state or behavior.

Demo

Cybage®
Delivering Value Scientifically

Responsibilities of Controllers and Services

Controllers	Services
Holds presentation logic	Holds business logic
Directly linked to a view	Independent of view
Drives the UI	Drives the application
Contains specific logic	Contains reusable logic
Responsible for taking decisions, such as what data to fetch, what data to show, how to handle user interaction, styling and display of UI.	Responsible for making server calls, common validation logic, application level stores, reusable business logic.

Cybage®
Delivering Value Scientifically

Dependency Injection in AngularJS

What is Dependency Injection

- Instead of creating instances of dependent service, a function should request one when needed
- Angular will be responsible for creating an instance of dependent service and pass/inject it to the function

Benefits of Dependency Injection

- Reusability
- Modularity
- Testability

```
// Without Dependency Injection
function fetchDashboardData() {
  var $http  = new HttpService();
  return $http.get('my/url');
}

// With Dependency Injection
function fetchDashboardData($http) {
  return $http.get('my/url');
}
```

Cybage®
Delivering Value Scientifically

Dependency Injection in AngularJS

Safe style of dependency injection

Declaring as string

Injecting as a variable

- **myModule.controller('MainCtrl', ['$log', function($log){ }]);**

Order of the injection also depends on this string declaration.

The other syntax

- **myModule.controller('MainCtrl', function($log){ });**

Can be renamed to something else such as x/y/z during Minification. In that case angular will not be able to figure out what service we need.

Cybage®
Delivering Value Scientifically

Built-in Services in AngularJS

- Angular uses $ prefix for all the services in the library
- $window
  - A wrapper around the global window object
  - Used to avoid global state during tests
  - In unit test it can be mocked out using AngularJS mocking library
- $location
  - Allows to interact with the urls in browser's address bar
  - Get and manipulate its value
  - Any changes made in $location is reflected in the browser
  - Any changes in the url is reflected in $location
- $http
  - Used to make XHR requests to the server from application
  - Can make GET and POST requests
  - Set the headers and caching
  - Deal with server responses and failures.

Cybage®
Delivering Value Scientifically

Creating a Custom Service in AngularJS

- Custom services should not include $ sign
- Use the angular.module().factory function to declare the service's name and dependencies
- The service will be lazily instantiated.
  - The very first time a controller, service, or directive asks for the service, it will be created.
  - The service definition function will be called once, and the instance stored.
  - Every caller of this service will get this same, singleton instance handed to them

- **Note** : Singleton instance is important because in a Single Page Application, the HTML and controllers can get destroyed and created multiple times in an application.

Different ways to create services

- Factory
  - When following functional style of programming
  - When you want to return function and objects
- Service
  - When we follow object oriented programming
  - Angular calls new on the function to create instance
- Provider
  - When we need to set up configuration for our service
  - Sets up how services should work based on environment.

Demo

$http Service

```
angular.module('notesApp', [])
  .controller('MainCtrl', ['$http', function($http) {
    var self = this;
    self.items = [];
    $http.get('/api/note').then(function(response) {
      self.items = response.data;
    }, function(errResponse) {
      console.error('Error while fetching notes');
    });
  }]);
```

- A core Angular service to communicate with servers
- Used to make XHR (XmlHttpRequest) requests
- XHRs are asynchronous method calls(response time is unknown)
- AngularJS XHR API follows Promise interface
- The Promise interface guarantees how to handle those responses.

The Promise API

```
angular.module('notesApp', [])
  .controller('MainCtrl', ['$http', function($http) {
    var self = this;
    self.items = [];
    $http.get('/api/note').then(function(response) {
      self.items = response.data;
    }, function(errResponse) {
      console.error('Error while fetching notes');
    });
  }]);
```

- $http.get() returns a Promise object
- What we do with the Promise object
  - When server returns response(success/error) we can call a function
- The then() function takes two arguments
  - A success handler – called when response is 200
  - An error handler – called when response is non-200

Cybage® Delivering Value Scientifically

The Promise API

```
$http.get('/api/server-config').then(function(configResponse) {
    return $http.get('/api/' + configResponse.data.USER_END_POINT);
}).then(function(userResponse) {
    return $http.get('/api/' + userResponse.data.id + '/items');
}).then(function(itemResponse) {
    // Display items here
  }, function(error) {
    // Common error handling
});
```

- Each asynchronous task returns a promise object
- Each promise object will have a then function
  - The then function takes two arguments – a success or error handler
- The then function can also return a promise
  - To create a chaining of multiple server calls
- Each handler(success or error) can return a value
  - Which will be passed to the next function in the chain of promises
- If a handler returns a promise
  - Then the next handler will be called only after the request is finished.
- Common error handling function is called
  - In case of error anywhere in the promise chain.

Demo

AngularJS Filters

Cybage®
Delivering Value Scientifically

What are Filters in Angular

- Filters are used to format data and present to the user
- They can be applied to an expression in HTML
- Can be used in controllers and services
- Converts data into user readable format
- For example, adding currency symbol to numbers
- Syntax to write filters are

```
{{ expression | filter }}
{{ expression | filter1 | filter2 }}
```

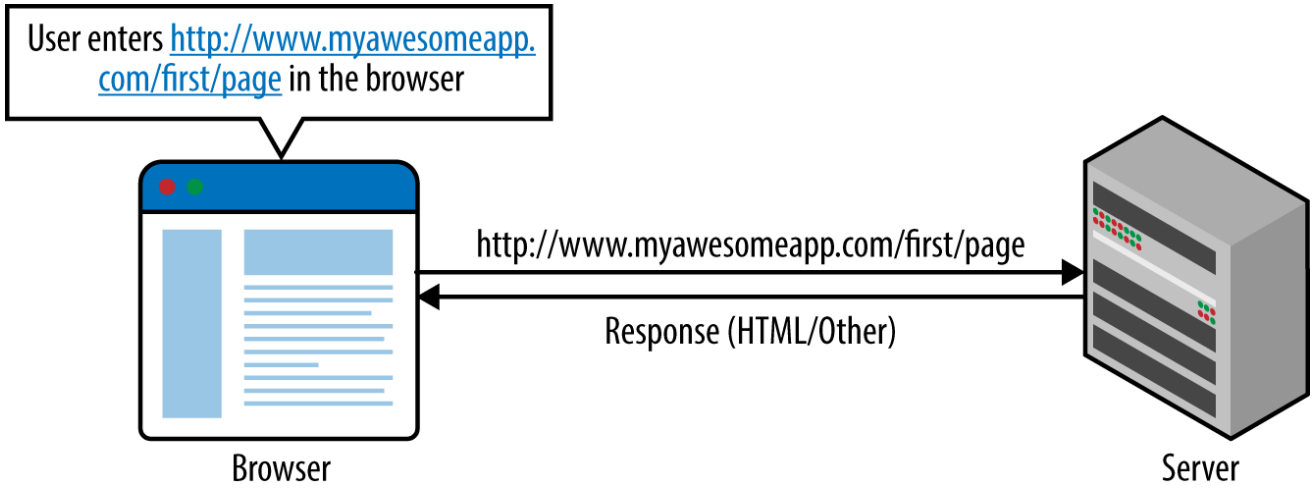
Common AngularJS Filters

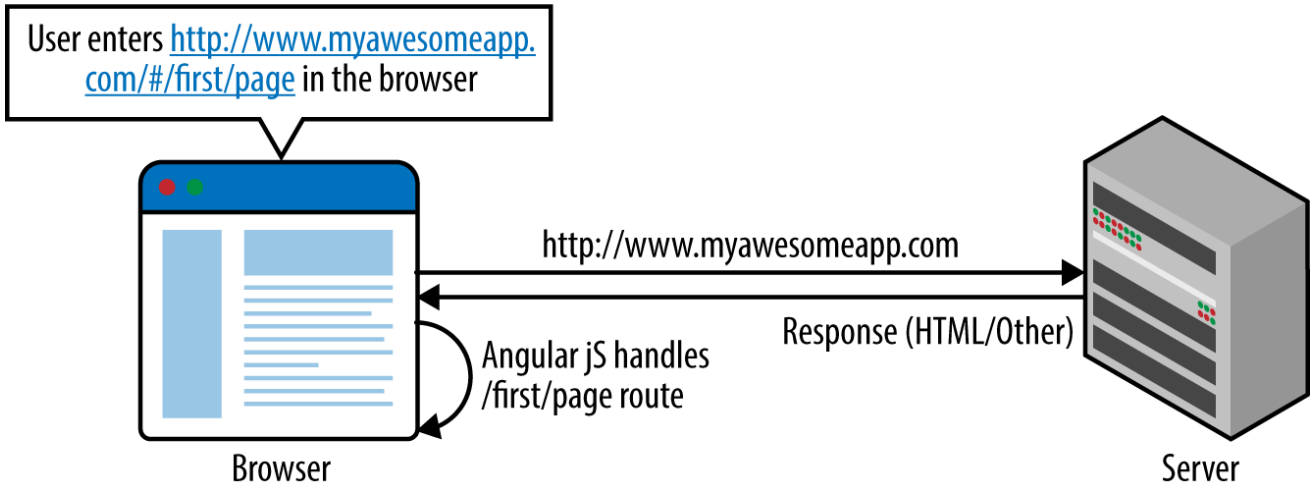
Filters	Description
currency	Formats a given number as a currency. Adds commas, decimals and currency symbols as needed.
number	The number filter takes a number and converts it to a human-readable string with comma separation.
lowercase	A very simple string filter that takes any string and converts all the characters to lowercase.
uppercase	A very simple string filter that takes any string and converts all the characters to uppercase.
json	It takes a JSON object or array and displays it as a string in the UI.
date	It takes a date object or a long timestamp and displays it as a human-readable string in the UI.

Demo

Routing in Angular

Routing in SPA





Angular Routing module

- AngularJS routing is optional module
- The library need to be included in the application.

Adding AngularJS routing library

```
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.11/angular-route.js">
</script>
```

Using ngRoute

Include the module as a dependent module into our main module

```
angular.module('routingApp', ['ngRoute'])
```

ng-view

Marking which section of page AngularJS should change when route changes.

```
<div ng-view></div>
```

$routeProvider

Defining routes in the config section using $routeProvider service.

```
angular.module('routingApp', ['ngRoute'])
  .config(['$routeProvider', function($routeProvider) {

    $routeProvider.when('/', {
      template: '<h5>This is the default route</h5>'
    })
    .when('/second', {
      template: '<h5>This is the second route</h5>'
    })
    .otherwise({redirectTo: '/'});
  }]);
```

Cybage®
Delivering Value Scientifically

when function

The when function takes two parameters. The first is URL and second is a configuration object which knows what to do when this route is encountered.

```
angular.module('routingA        ['ngRoute'])
  .config(['$routeProvi        ', function($routeProvider) {

    $routeProvider.when('/', {
      template: '<h5>This is the default route</h5>'
    })
    .when('/second', {
      template: '<h5>This is the second route</h5>'
    })
    .otherwise({redirectTo: '/'});
  }]);
```

Cybage®
Delivering Value Scientifically

otherwise function

The otherwise function knows what to do if user wants to go to a URL which is not known to AngularJS or not specified in the configuration.

```
angular.module('routingA        ['ngRoute'])
  .config(['$routeProvi        ', function($routeProvider) {

    $routeProvider.when('/', {
      template: '<h5>This is the default route</h5>'
    })
    .when('/second', {
      template: '<h5>This is the second route</h5>'
    })
    .otherwise({redirectTo: '/'});
  }]);
```

Demo

Unit Testing

What is Unit Test

- A concept of checking a part of code/a function
- Writing assertions and tests to check the function
- To make sure the function works as intended
- Common in server-side but rare in client-side.

Why Unit Test

- Proof of correctness
  - Make sure the code delivers correct results in all cases
- Lack of compiler
  - JavaScript doesn't have a compiler like Java/.NET
  - Code produces different results in different browsers
  - Unit tests can catch problems in code well before it runs in the browser

What is TDD

- Test-driven development is an AGILE methodology
- It flips the development life cycle
- The tests are written before the code is implemented
- The tests drive the application development.

Karma – A Test Runner

karma-runner.github.io/1.0/index.html

KARMA

intro config plus dev about v1.0

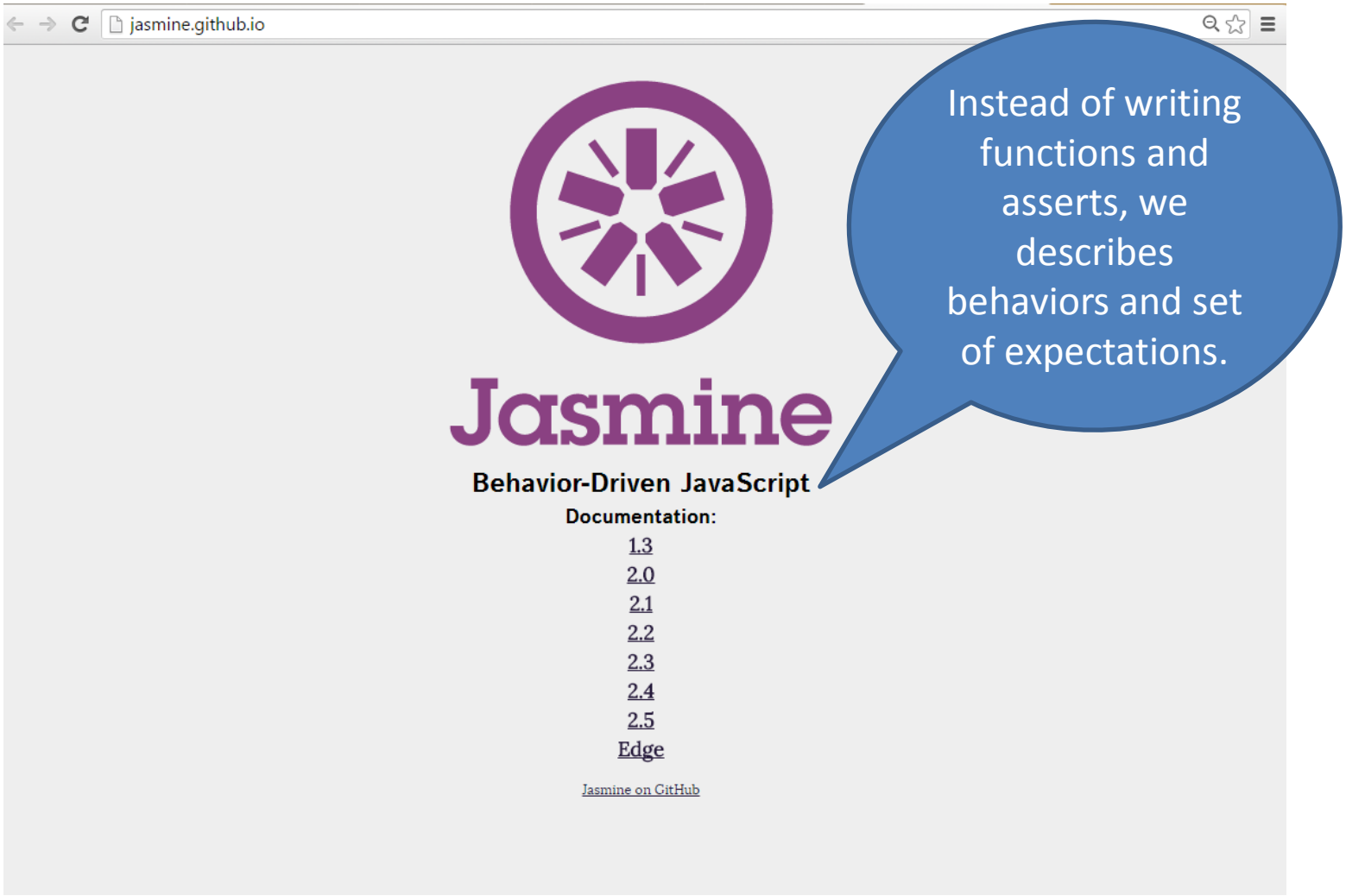
On the AngularJS team, we rely on testing and we always seek better tools to make our life easier. That's why we created Karma - a test runner that fits all our needs.

View project on GitHub

npm install karma

www.youtube.com/watch?v=5mHjJ4xf_K0

Jasmine – A Testing Framework



Cybage®
Delivering Value Scientifically

Writing Tests – Example1

```
// A Test Suite in Jasmine
describe('My Function', function() {

  var t;
  // Similar to setup
  beforeEach(function() {
    t = true;
  });

  afterEach(function() {
    t = null;
  });

  it('should perform action 1', function() {
    expect(t).toBeTruthy();
  });
  it('should perform action 2', function() {
    var expectedValue = true;
    expect(t).toEqual(expectedValue);
  });
});
```

Cybage®
Delivering Value Scientifically

Writing Test – Example2

```
describe('Controller: ListCtrl', function() {
  // Instantiate a new version of my module before each test
  beforeEach(module('notesApp'));

  var ctrl;

  // Before each unit test, instantiate a new instance
  // of the controller
  beforeEach(inject(function($controller) {
    ctrl = $controller('ListCtrl');
  }));

  it('should have items available on load', function() {
    expect(ctrl.items).toEqual([
      {id: 1, label: 'First', done: true},
      {id: 2, label: 'Second', done: false}
    ]);
  });
```

Cybage®
Delivering Value Scientifically

Summary

- Introduction
- SPA
- MVC Pattern
- Module
- Controller
- Template
- Data Binding
- Directives
- Forms
- Services
- Filter
- Routing
- Unit Test

Bibliography, Important Links

https://docs.angularjs.org/guide

Any Questions?

Thank you!