AngularJS

1. Defer & Promise Objects
2. Communication between Controllers
3. Controller Inheritance
4. Internationalization
5. Application Folder Structure
6. Custom Directives

Defer & Promise Objects

There are two objects to

1) keep track the status of task or asynchronous operation.

2) Synchronize multiple asynchronous calls

keep track the status of task or asynchronous operation.

Deferred object:

Deferred object is an object which represents a task that will finish in the future.

Promise Object:

A promise is object which represents *the eventual result of an asynchronous operation.*

Steps to track the status of task or asynchronous operation.

1) Create the defer object by using the $q services

```
var deferred = $q.defer();
```

2) Attach a task to deferred object to track status

```
deferred.reject(error result);

deferred.resolve(success response);
```

3) Send promise object to consumer who ever wants this task. This object sends immediately to the consumer irrespective of result of the task.

4) In consumer side, receive the promise object and call the callback method to know the status

```
promiseObject.then(function(rest) {
    //Receives the response
  })
.catch(function(rest) {
    //Receives the error response
  })
```

Synchronize multiple asynchronous calls

$q.all() can be used to synchronize the multiple synchronous calls

Communication between Controllers

```
<div ng-controller="cntrl1">  //Publishes

<span>{{msg1}}</span>

</div>


<div ng-controller="cntrl2">  //Subscribes

<span>{{msg2}}</span>

</div>
```

$broadcast

It dispatches an event to all child scopes and notify to the registered Scope listeners. All listeners for the event on this scope get notified.

```
    $rootscope.$broadcast('eventName', eventdata);
```

$on

It listen on events of a given type. It can catch the event dispatched by $broadcast.

```
$scope.$on('eventName', function(eventDetails,eventData) { });
```

Controller Inheritance

Scope are controllers specific. If we defines nested controllers then child controller will inherit the scope of its parent controller

```
<div ng-controller=”ParentController”>
        {{msg1}}
            <div ng-controller=”ChildController1”>
            </div>
            <div ng-controller=”ChildController2”>
            </div>
</div>
```

Internationalization:

Many web applications need to support multiple languages.

Internationalization (i18n) - Making your application able to support a range of languages and locales.

•**Language** - For example, Spanish generally. ISO code "en".

•**Locale** – language+country ex:en_US,fr_FR,de_DE

Steps to build Internationalization application

1) Include the below dependencies

angular-translate.min.js

angular-translate-loader-static-files.min.js

2) Create translation files(which are in .json format) which contain the messages to be localized and update keys at HTML file

```
<h2 translate="SUBHEADER"></h2>
```

3) Configure the dependency module for our module

```
var myAppModule = angular.module('myApp', [ 'pascalprecht.translate' ]);
```

4) Load translation files and set default language in configuration phase(Application startup phase)

```
myAppModule.config( function($translateProvider) {
                                    $translateProvider.useStaticFilesLoader( {
```

```
                                prefix : 'l18n/',

                                suffix : '.json'

                        });

                        // Tell the module what language to use by default

                        $translateProvider.preferredLanguage('en_US');

    } )
```

5) Set the locale dynamically in our controller

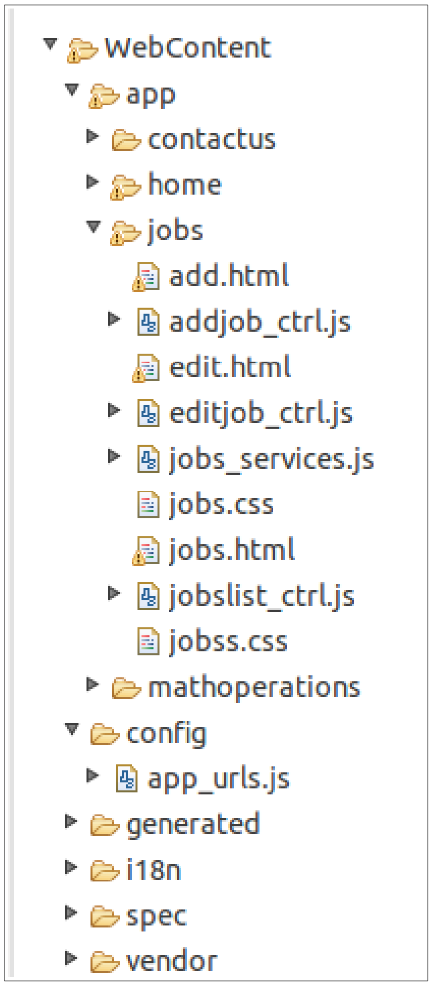
```
 $translate.use(langKey/locale);
```

Currency & Date symbols display

Displays the currencies and Dates as per locale. We need to include the below dependency as per selected locale

```
<script src="https://code.angularjs.org/1.2.5/i18n/angular-locale_fr-ch.js"></script>
```

Application Folder Structure



```
▼ WebContent
  ▼ app
    ▶ contactus
    ▶ home
    ▼ jobs
        add.html
      ▶ addjob_ctrl.js
        edit.html
      ▶ editjob_ctrl.js
      ▶ jobs_services.js
        jobs.css
        jobs.html
      ▶ jobslist_ctrl.js
        jobss.css
    ▶ mathoperations
  ▼ config
    ▶ app_urls.js
  ▶ generated
  ▶ i18n
  ▶ spec
  ▶ vendor
```

Custom Directives

AngularJS directives controls the rendering of the HTML elements inside an AngularJS application. Examples of predefined directives are the interpolation directive ({{ }}), the ng-repeat directive and ng-if directive.

It is possible to implement your own directives too. .

Directive Types

You can implement the following types of directives:

- Element directives
- Attribute directives

A Basic Directive

You register a **directive** with a module. Here is an example of how that looks:

```
myapp = angular.module("myapp", []);

myapp.directive('firstDirective', function() {

    var directive = {};

    directive.restrict = 'EA'; /* restrict this directive to elements */

    directive.template = "<span>My first directive</span>";

    return directive;

});

<div firstDirective/>

<div>Sample text</div>
```