Angular in examples

by Michał Taborowski

Agenda

- What is Angular
- Expressions
- Model and binding
- Forms
- Dependency Injection
- Controllers
- Scopes
- Views and templates
- Directives
- Promises
- XHR
- Routing
- UI Tools – bower, grunt, closure compiler

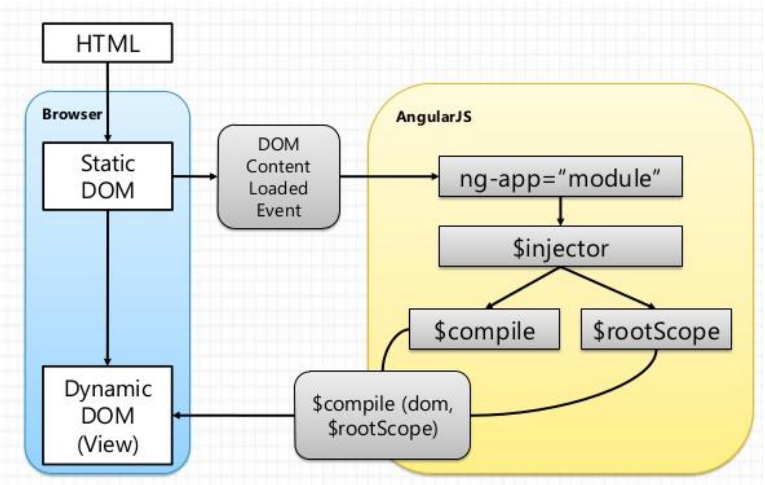
What is Angular ?

MVC client-side web application framework

Lets extend HTML syntax

Automatically synchronizes UI with object (2-way binding)

Uses dependency injection



Application structure

Folder layout

- angular-seed – type-centric layout
- Google-style – feature-centric layout
- **GTech-style – the best**

Simplest angular application

```
angular.module('app', [])
    .config(function ($providers,$constant) {
    })
    .run(function ($service,$factory,$value) {
    });
```

Expressions (http://jsfiddle.net/michal_taborowski/54BHe/)

Angular evaluates everything between {{ and }}, so :

```
<div ng-app>
  {{('Hello ' + 'world').indexOf('e')}}
</div>
```

evaluates to :

```
1
```

Executed in context of current scope – have access to local `$scope` variable

Simple expresions - no flow control syntax

Expressions – part 2

{{ and }} are configured in `$interpolateProvider` with `startsSymbol` and `endSymbol` – can set your own

We can evaluate expresion in js with `$iterpolate` service

```
var template = $interpolate(text);
return template(context);
```

Filters (http://jsfiddle.net/michal_taborowski/yUT6L/)

Used to apply formatting or filtering, can be used in the views as well in the controllers, services and directives – we can inject `$filter` service

Built in filters :

filter, currency, number, date, json, lowercase, uppercase, limitTo, orderBy

```
{{ 3 | number:2 }}  == $filter(”number”)(3,2) = 3.00
```

Be carful when using in a view, can lower performance when applied to a every element of an array

Model and binding (http://jsfiddle.net/michal_taborowski/4DLyL/)

Model - Plain Javascript object

Use `ng-model` directive

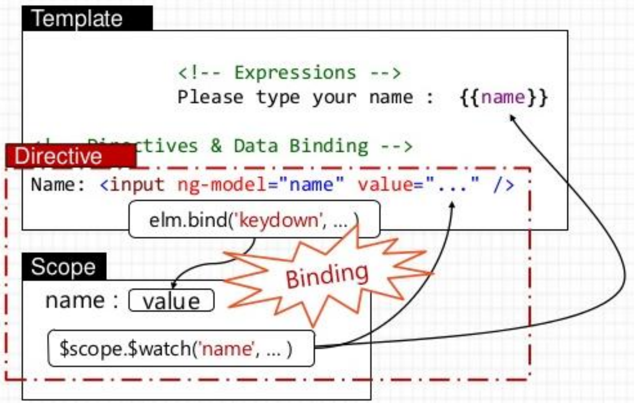
`$viewValue` – updated by the view

`$modelValue` – held by model

2-way binding – model updates the view and view updates model

1-way binging - improves performance (:: operator)

Under the hood :



```
<!-- Expressions -->
Please type your name :  {{name}}

<!-- Directives & Data Binding -->
Name: <input ng-model="name" value="..." />
```

Forms (http://jsfiddle.net/michal_taborowski/3f48skL3/)

from directive – not plain html tag! – adds controller with validation etc.

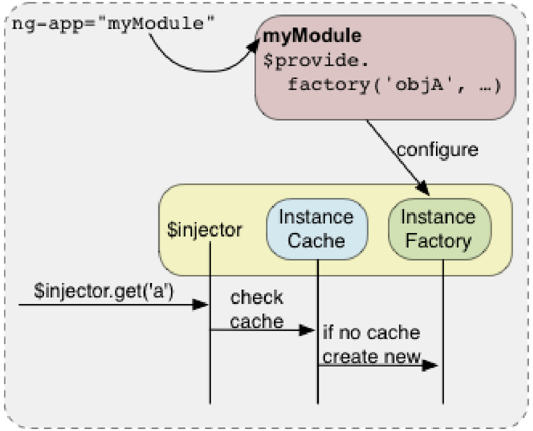
Validation states (properties of the form and the fields):

`$pristine, $dirty, $valid, $invalid`

`form.field.$invalid`

- `form.field.$error.validator`
  - *built in validators : min, max, length, required, email,pattern etc.*
  - *can write own and set validity with ngModelCtrl.$setInvalid('validator',true |false)*

Dependency injection (http://jsfiddle.net/michal_taborowski/U42rs/)



Key feature of Angular

Definition `$provide`, usage `$injector`

Types of $providers

provider, constant, value, factory, service

Can get any provider by calling `$injector.get()`

Can inject into function with `$injector.invoke()`

Instances are created only once, result of provider is cached

Every service has its provider ie. `userService` has `userServiceProvider`

Only providers can be injected into `config` phase, except constant

Used to configure injectables

Dependency injection – part 2

factory

```
angular.module('myApp',[]).factory('myFactory', function(){
       return { sayHello : function(){ return "Hello world"} }
    }
)
```

service

```
angular.module('myApp',[]).service('myService', function(){
        this.sayHello = function(){ return "Hello world"}
    }
)
```

value

```
angular.module('myApp',[]).value('userConfig',{ language : 'en', currency : 'USD'})
```

provider

```
angular.module('myApp',[]).provider('myProvider', function(){
       this.$get = function(){
         return {
           sayHello : function(){ return "Hello world"}
         }
       }
    }
)
```

constant

```
angular.module('myApp',[]).constant('TOKEN','MySecretToken')
```

Dependency injection – part 3

You cannot inject controllers – seems logic

`$controller` service, `$controllerProvider`

Filters - `$filter` service, `$filterProvider`

Directives - `$compile` service, `$compilerProvider`

Injection

```
myApp.controller('Ctrl',function($injectables){});

myApp.controller('Ctrl',['a$injectables',function($injectables){}])

var Ctrl = function(a$Injectables){}
Ctrl['$inject'] = ['$injectables']
```

Controllers (http://jsfiddle.net/michal_taborowski/vH66L/)

- Place to call business logic
- Standard syntax :

```
angular.module('app',[]).controller('Ctrl',['$scope',function($scope){
   $scope.data = { message : 'This is message'};
}])
```

```
<div ng-controller="Ctrl">
  <span>{{ data.message }}</span>
</div>
```

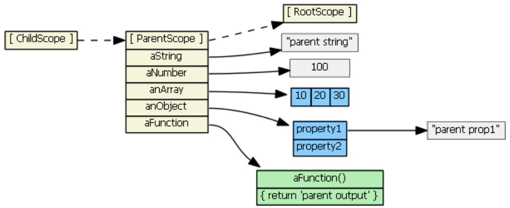
- controller-As syntax

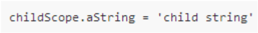
```
angular.module('app', []).controller('Ctrl', [function (){
   this.data = {message: 'This is message'}
}])
```

```
<div ng-controller="Ctrl as ctrl">
  <span>{{ ctrl.data.message }}</span>
</div>
```

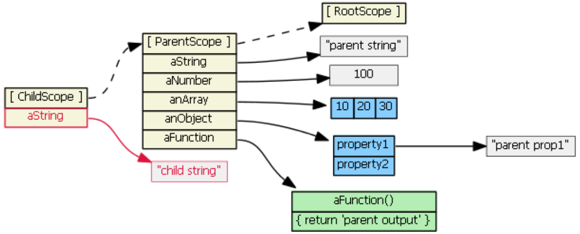
Scopes

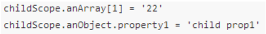
- „Link” between model and template
- Scope != model , scope has reference to the model (model – POJO)
- Scopes (child-parent) uses JavaScript Prototypal Inheritance



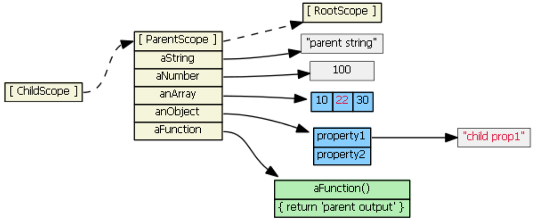


```
childScope.aString = 'child string'
```





```
childScope.anArray[1] = '22'
childScope.anObject.property1 = 'child prop1'
```



Scopes part 2 – child scopes (http://jsfiddle.net/michal_taborowski/Da3xZ/)

- one `$rootScope` per application, but many child scopes
- The following create new scopes, and inherit prototypically: `ng-repeat`, `ng-include`, `ng-switch`, `ng-view`, `ng-controller`, directive with scope: true, directive with transclude: true.
- 2 way-binding with primitives :

  Dot rule – if you don't have dot in your `ng-model`, probably you are doing it wrong!

Scopes part 3 – Angular event loop

(http://jsfiddle.net/michal_taborowski/2z5KX/)

- `$digest()`

  dirty checking of bindings

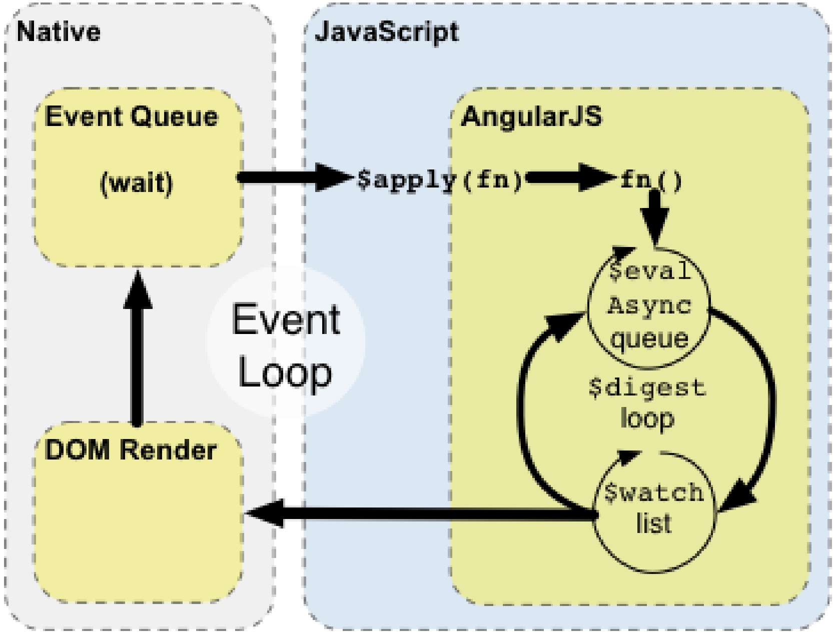
  should not be called directly

- `$apply()`

  enters angular context

  enables integration with 3-rd party libs

  used by Angular ie. `ng-click`, `ng-model`, `$http` callback



Scopes part 4 – watching and events

(http://jsfiddle.net/michal_taborowski/ee2DG/)

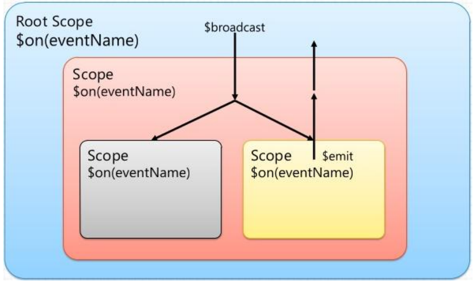
API to `$watch/$watchCollection/$watchGroup` model modifications

- `$watch(watchExpression,[listener],[objectEquality])` – listener can change model again and again and again and … but only 10 times (internal Angular limit for $digest method)

Build-in events:

- $locationChangeStart, $locationChangeSuccess, $routeChangeStart, $routeChangeSuccess, $routeChangeError, $routeUpdate, $estroy, $includeContentLoaded, $includeContentRequested, $viewContentLoaded

Can `$emit` and `$broadcast` own events and process them with `$on` function



Views and templates (http://jsfiddle.net/michal_taborowski/BNS3H/)

HTML with Angular specific expressions and directives

Declare with :

```
<script type="text/ng-template" id="template.html"> </script>
```

Include with :

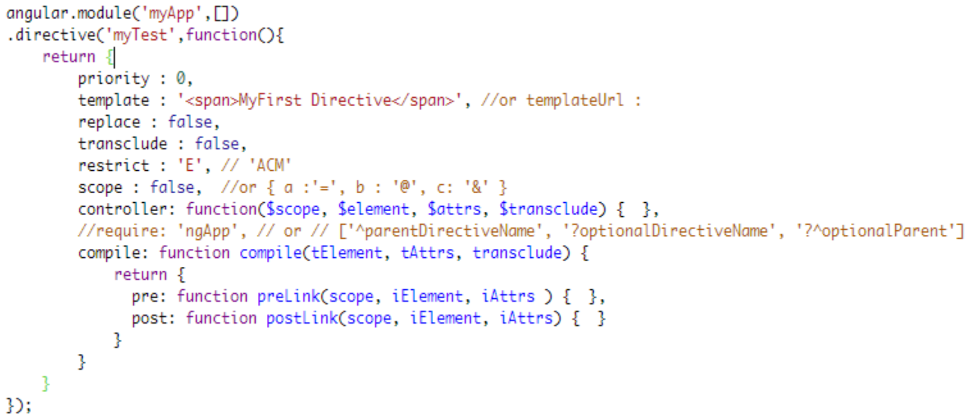
```
<div ng-include=" 'template.html' " />
```

Cached for performance, available with `$templateCache`

Directives (http://jsfiddle.net/michal_taborowski/vYTa6/)

- Additional marker on a DOM compiled by Angular
  element (E), attribute (A), CSS class(C), comment (M)
- The only place that should manipulate DOM
- Built-in like : `ngModel`, `ngRepeat`, `ngIf`, `ngShow`, `ngClass`,
- Directive template

  (Directive Definition Object) :



```
angular.module('myApp',[])
.directive('myTest',function(){
    return {
        priority : 0,
        template : '<span>MyFirst Directive</span>', //or templateUrl :
        replace : false,
        transclude : false,
        restrict : 'E', // 'ACM'
        scope : false,  //or { a :'=', b : '@', c: '&' }
        controller: function($scope, $element, $attrs, $transclude) {  },
        //require: 'ngApp', // or // ['^parentDirectiveName', '?optionalDirectiveName', '?^optionalParent']
        compile: function compile(tElement, tAttrs, transclude) {
            return {
              pre: function preLink(scope, iElement, iAttrs ) {  },
              post: function postLink(scope, iElement, iAttrs) {  }
            }
        }
    }
});
```

Directives – part 2

link - function(scope, element, attrs, contoller), links scopes with bindings

- Injecting required controller :
  - *? For conditional (e.g. ‚?ngModel’) or null*
  - *^ search in parent scope*

ccontroller – standard controller with dependencies. Cannot access other directives controllers (e.g. ng-model)

Scope

- Inherited - `scope : true`
- Isolated - `scope : {`

```
element1 : ’@attribute’, // interpalated
element2 : ’=attribute’, // two-way binding
element3 : ’&attribute’  // callback
}
```

$q (http://jsfiddle.net/michal_taborowski/5WFZt/ , http://jsfiddle.net/michal_taborowski/tb7xR/)

Promises

- API for result of asyc actions

```
$q.defer()
  .resolve()
  .reject()
  .notify()
```

- Used heavily with `$http`
- Can resolve many requests at once with `$q.all`

XHR – part 1 $http (http://jsfiddle.net/michal_taborowski/M8R3D/)

`$http` – low level HTTP communication service

```
$http({method: 'GET', url: '/someUrl'})
     .success(function(data, status, headers, config) {
       //async promise
     })
     .error(function(data, status, headers, config) {
       //async promise
     });

$http.get('/someUrl').success(successCallback); // .then(function(){});
$http.post('/someUrl', data).success(successCallback); // .then(function(){});
```

XHR – part 2 $resource

High level service for RESTful services

```
var obj = $resource('/user/:id',null,
{ 'get': {method:'GET'},
  'save': {method:'POST'},
  'query': {method:'GET', isArray:true},
  'remove': {method:'DELETE'},
  'delete': {method:'DELETE'}
}).get(function(){
    obj.someProperty = newValue;
    obj.$save();

});
```

XHR – part 3 interceptors (http://jsfiddle.net/michal_taborowski/VW2QJ/)

`$httpProvider.interceptors` array of pre-request and post-response action

- `request`
- `response`
- `requestError`
- `responseError`

used for i.e. loading indicators, logging, authorization

Routing – part 1 ngRoute (http://jsfiddle.net/michal_taborowski/B77vW/)

Defined in config phase

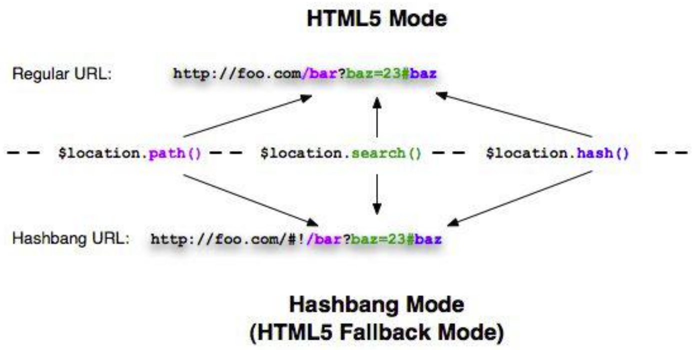
```
$routeProvider
    .when(url)
    .otherwise()
```

```
$locationProvider.html5Mode(true);
$locationProvider.hashPrefix('!');
```

Use $location service to navigate

```
.url()
.path()
.search()
.hash()
```

etc…



Routing – part 2 ui-router (http://jsfiddle.net/michal_taborowski/tQX9Q/)

- External module
- Allows nested views and many views on one page
- Based on states not only on URL’s

Testing

Unit tests – Jasmin + Karma

End-to-end tests - Protractor

Additional stuff

closure compiler

bower

gulp/grunt

Resources

- Angular - https://angularjs.org/
  - Wiki - https://github.com/angular/angular.js/wiki
  - Modules - http://ngmodules.org/
- Angular-ui - http://angular-ui.github.io/
  - UI-Router
  - UI-Bootstrap
- egghead.io - https://egghead.io/technologies/angularjs
- Youtube - https://www.youtube.com/user/angularjs
  - Best Prectices- https://www.youtube.com/watch?v=ZhfUv0spHCY&list=LLbn1OgGei-DV7aSRo_HaAiw - Must see !

Jsfiddles

Expressions http://jsfiddle.net/michal_taborowski/54BHe/

2 way binding http://jsfiddle.net/michal_taborowski/4DLyL/

Child scope http://jsfiddle.net/michal_taborowski/Da3xZ/

Ctrl module http://jsfiddle.net/michal_taborowski/vH66L/

Filters http://jsfiddle.net/michal_taborowski/yUT6L/

Templates http://jsfiddle.net/michal_taborowski/BNS3H/

Events http://jsfiddle.net/michal_taborowski/ee2DG/

Directives http://jsfiddle.net/michal_taborowski/vYTa6/

DI http://jsfiddle.net/michal_taborowski/U42rs/

$http http://jsfiddle.net/michal_taborowski/M8R3D/

$digest $apply http://jsfiddle.net/michal_taborowski/2z5KX/

$q http://jsfiddle.net/michal_taborowski/5WFZt/

$q.all & $http http://jsfiddle.net/michal_taborowski/tb7xR/

ngRoute http://jsfiddle.net/michal_taborowski/B77vW/

Ui-router http://jsfiddle.net/michal_taborowski/tQX9Q/ | http://jsfiddle.net/michal_taborowski/vAMfT/

$http & interceptors http://jsfiddle.net/michal_taborowski/VW2QJ/