Week 4

coursera.org/learn/single-page-web-apps-with-angularjs/discussions/weeks/4/threads/rO0C2hUgEeeskRI8P5CzrA

** This is a review from **Lecture 36** until **Lecture 41** (split in 2 parts)*

AngularJS – Routing with ui-router (Part 2/2)

(continuing...)

9) Declare **controllers** straight in the states: Pull down the controller declarations from all templates and declare them inside the states:

a) Add controllers to the states:

```
.state('home', {
  url: '/',
  templateUrl: 'src/templates/home.html',
  controller: 'HomeController as home'
})
.state('menu', {
  url: '/menu',
  templateUrl: 'src/app/templates/menu.html',
  controller: 'MenuController as menu'
});
```

b) Remove ng-controller from menu.html template:

```
<div id="menu">
  <a ui-sref="home" ui-sref-active="activeState">Home</a> &lt; <span
      >Menu</span>
  <h3>Items</h3>
  <menu-items items="menu.items"></menu-items>
</div>
```

10) Use **resolve** property to ensure the data before switching to a view.

a) Inside resolve's propery object, define a key (**items**) to resolve to a promise that gets returned by some service (**MenuService**). For that, declare an array to protect the service injection from minification:

```
.state('menu', {
  url: '/menu',
  templateUrl: 'src/app/templates/menu.html',
  controller: 'MenuController as menu',
  resolve: {
    items: ['MenuService', function (MenuService) {
      return MenuService.getItems();
    }]
  }
});
```

> When resolve property is a promise, the **transition** to the state only happens after it is resolved. Router **does not transition** if promise is rejected.

b) Inject / pass the resolved data into the controller function declaration:

```
(function () {
'use strict';
  angular.module('App')
    .controller('MenuController', MenuController);
  MenuController.$inject = ['items'];
  function MenuController(items) {
    var menu = this;
    menu.items = items;
  }
})();
```

11) Set up Routing State with URL Parameters: Parameters are wrapped in curly braces and can have complex matching rules such as regular expression. Use $stateParams service for retrieving parameters

```
...
  .state('item', {
    url: '/item/{itemID}',
    templateUrl: 'src/app/templates/item.html',
    controller: 'ItemController as item',
    resolve: {
      item: ['$stateParams', 'MenuService', function ($stateParams,
          MenuService) {
        return MenuService.getItems().then( function (items) {
          return items[$stateParams.itemID];
        });
      }]
    }
  });
...
```

P.S.: If you do not want to work with URL, provide **params: { itemID: null }** instead. This tells ui-router to expect a param internally:

```
...
  .state('item', {
    params: { itemID: null },
    templateUrl: 'src/app/templates/item.html',
    ...
```

Example for the template *item.html*:

```
<div>Item name: {{ item.name }}</div>
<div>Item quantity: {{ item.qtty }}</div>
<div>Item description: {{ item.desc }}</div>
```

Example for ItemController ('item' is here being passed as 'itemParam'):

```
(function () {
'use strict';
  angular.module('App')
    .controller('ItemController', ItemController);
  ItemController.$inject = ['itemParam'];
  function ItemController(itemParam) {
    var item = this;
    item.name = itemParam.name;
    item.qtty =  itemParam.qtty;
    item.desc =  itemParam.desc;
  }
})();
```

12) Construct a URL passing param with **ui-sref** directive

Following format: ui-sref="stateName({paramName: value})

a) Make menu item clickable by adding **ui-sref** in <**li**> element in the menu template:

```
<ul>
  <li ng-repeat="item in $ctrl.items" ui-sref="item({itemID: $index}
      )">
    {{ item.qtty }} of {{ item.name }}
  </li>
</ul>
```

b) And give it some style (*styles.css*):

```
li[ui-sref]:hover {
  display: inline-block;
  cursor: pointer;
  border: solid black 1px;
  padding: 0 5px 0 5px;
}
```

13) Nested States are nested views: To set up a nested (child) state, declare a state with its name preceeded by another state's name (the parent scope for this state) and a dot

('parent.child'). This configuration requires a new declaration of the ui-view (<ui-view></ui-view>) in the parent's template for the child's template to be inserted.

a) Change the '**item**' state by preceedig its name with '**menu**.' and removing the resolve property altogether. The resolved **items** from 'menu' state will be automatically passed into (inherited by) our new nested 'menu**.item**' state, which also goes into the child's controller ItemController (becomes injectable):

```
...
  .state('menu.item', {
    url: '/item/{itemID}',
    templateUrl: 'src/app/templates/item.html',
    controller: 'ItemController as item'
  });
...
```

> This optional child URL gets concatenated with the parent's and will show up in the browser as ...#/menu/item/

b) Change ItemController by injecting '$stateParams', replacing **itemParam** for **items** in the injection and in the function's argument, and creating new var itemParam and assigning its value from items[$stateParams.itemID]:

```
...
  ItemController.$inject = ['$stateParams', 'items'];
  function ItemController($stateParams, items) {
    var item = this;
    var itemParam = items[$stateParams.itemID];
    item.name = itemParam.name;
    item.qtty =  itemParam.qtty;
    item.desc =  itemParam.desc;
  }
...
```

c) Update the new nested state's name in **ui-sref** in the *items.html* template:

```
<ul>
  <li ng-repeat="item in $ctrl.items" ui-sref="menu.items({itemID:
      $index})">
    {{ item.qtty }} of {{ item.name }}
  </li>
</ul>
```

d) Add ui-view directive to *menu.html* template for the child to be inserted:

```
<div id="menu">
  <a ui-sref="home">Home</a> &lt; <span>Menu</span>
  <h3>Items</h3>
  <menu-items items="menu.items"></menu-items>
  <ui-view></ui-view>
</div>
```

14) State Transition Events: This short example uses ui-router state change events to show and hide a spinner by firing events on the $rootScope. $stateChangeStart starts a state transition; $stateChangeSuccess tells if a transition ended successfully; $stateChangeError tells whether a transition has failed and whether there was any errors in the resolve (erros from the resolve do not display on the Console).

```
(function () {
'use strict';
  angular.module('App', ['ui.router', 'Spinner']);
})();
(function () {
'use strict';
  angular.module('Spinner', []);
})();
(function () {
'use strict';
  angular.module('Spinner')
    .component('loading', {
      templateUrl: 'src/spinner/template/loading.html',
```

```
      controller: SpinnerController
    });
    SpinnerController.$inject = ['$rootScope']
    function SpinnerController($rootScope) {
      var $ctrl = this;
      var listeners = [];
      $ctrl.$onInit = function () {
        var cancel  = $rootScope.$on('$stateChangeStart', function
            (event, toState, toParams, fromState, fromParams,
            options) {
          $ctrl.show = true
        });
        listeners.push(cancel);
        cancel = $rootScope.$on('$stateChangeSuccess', function
            (event, toState, toParams, fromState, fromParams) {
          $ctrl.show = false;
        });
        listeners.push(cancel);
        cancel = $rootScope.$on('$stateChangeError', function(event,
            toState, toParams, fromState, fromParams, error) {
          $ctrl.show = false;
        });
        listeners.push(cancel);
      };
      $ctrl.$onDestroy = function () {
        listeners.forEach(function (item) {
          item();
      });
    };
})();
```

```
<img ng-if="$ctrl.show" class="loading" src="img/loading.gif" alt
    ="loading">
```

*** End of Part 2 of 2 ***