ANGULAR UI-ROUTER

A STATE MANAGER FOR THE MASSES

Created by Adam Grant

PROBLEMS WITH ANGULAR ROUTER

- Single view - nesting is a PITA!
- Updating page "widgets" requires shared scope or service
- Checking for ancestry on route change is difficult (Regex?)

WHY IS UI-ROUTER AWESOME?

- Multiple views
- Nestable states
- Views ~ States != 1:1

EXAMPLE

Default view (single unnamed)

```
<!-- in index.html -->
<body ng-controller="MainCtrl">
    <div ui-view></div>
</body>
```

Multiple Named Views

```
<!-- in index.html -->
<body ng-controller="MainCtrl">
    <div ui-view="sidebar"></div>
    <div ui-view></div>
    <div ui-view="footer"></div>
</body>
```

^^^ SUPER POWERFUL!!! ^^^

STATE HIERARCHY

- contacts
  - contacts.list
  - contacts.detail
    - contacts.detail.bio
      - contacts.detail.bio.work_history
    - contacts.detail.skills

$STATEPROVIDER

- A "state" is a place in the UI/Nav
- Describes how the UI looks and acts via Controller, Template, View properties
- State hierarchy: parent-child / nested states

I LIKE MY STATE RARE

```
app.config(function ($stateProvider) {
  $stateProvider.state("contacts", {
    template:   '<h1>{{title}}</h1>',
    controller: function ($scope, $state, myTitle) {
      $scope.title    = myTitle;
      $scope.division = $state.current.data.division;
      $scope.segment  = $state.current.data.segment;
    },
    resolve: { myTitle: 'My Contacts' },
    data: {
      division: "HR",
      segment:  "Compensation"
    }
    onEnter: function (myTitle) { if (myTitle) {...} },
    onExit:  function (myTitle) { if (myTitle) {...} }
  })
```

HANDY EVENTS

- $stateChangeStart - fired when the transition begins
- $stateChangeSuccess - fired once the state transition is complete
- $stateChangeError - fired when an error occurs during transition
- $stateNotFound - fired when a requested state cannot be found
- $viewContentLoading - fired once the view begins loading, before the DOM is rendered
- $viewContentLoaded - fired once the view is loaded, after the DOM is rendered

EVENT HANDLING

```
$scope.$on('$stateChangeSuccess',
    function (event, toState, toParams, fromState, fromParams) {
        $scope.changeSomeShit(toState);
    }
)
$scope.$on('$stateChangeError',
    function (event, toState, toParams, fromState, fromParams) {
        alert("You be crazy: ", fromState.name)
    }
)
```

NESTING

Child states denoted by a "."

```
$stateProvider
  .state('contacts', {})
  .state('contacts.list', {});
  .state('contacts.list.treeview', {});
```

or

```
$stateProvider
  .state('contacts', {})
  .state('list', {
    parent: 'contacts'
  });
```

Children states won't work without a parent

WHAT GETS INHERITED?

Children inherit: data, resolve

```
$stateProvider.state('parent', {
  resolve:{
    resA:  function(){ return {'value': 'A'}; }
  },
  controller: function($scope, resA){
    $scope.resA = resA.value;
  }
}).state('parent.child', {
  resolve:{
    resB: function(resA){ return {'value': resA.value + 'B'}; }
  },
  controller: function($scope, resA, resB){
    $scope.resA2 = resA.value;
    $scope.resB = resB.value;
  }
});
```

ABSTRACT STATES

```
$stateProvider
    .state('contacts', {
        abstract: true,
        url: '/contacts',              // Not actually Navigable!
        template: '<ui-view/>',
        controller: 'ContactsController'
    })
    .state('contacts.list', {
        url: '/list'                   //  url === "/contacts/list"
    })
    .state('contacts.detail', {
        url: '/detail',                //  url === "/contacts/detail"
        templateUrl: 'contacts.details.html'
    })
```

MULTIPLE VIEWS

```
<!-- in index.html -->
<body ng-controller="MainCtrl">
    <div ui-view="sidebar"></div>
    <div ui-view></div>
    <div ui-view="footer"></div>
</body>
```

```
<!-- in contacts.html -->
<div class="contacts">
    <h2>Contacts</h2>
    <div ui-view></div>
</div>
```

```
$stateProvider
  .state('contacts', {
    views: {
      'sidebar': {
        templateUrl: 'sidebar.html',
        controller: 'SideBarController'
      },
      '': {
        templateUrl: 'contacts.html',
        controller: 'ContactsController'
      },
      'footer': {
        templateUrl: 'footer.html'
      }
    }
  })
```

NESTING CRAZINESS!

```
$stateProvider
  .state('contacts.detail', {
    views: {
      'sidebar@contacts': {                    // Modifies parent sidebar UI
        templateUrl: 'contacts/sidebar.html', // at (ui-view=sidebar)
        controller: 'DetailSideBarController'
      },
      '': {                                    // Nests inside of
        templateUrl: 'contacts/detail.html',  // contacts.html
        controller: 'ContactsDetailController'
      },
      'footer@': {                             // Another way to replace
        templateUrl: 'contacts/footer.html'   // the top-level footer
      }                                        // (ui-view=footer)
    }
  })
```

Use the "@" sign to target ui-views in other states

Something like "nameOfUiView@" will target top level state's template

TARGETING STATES

View	UI View	State
footer@contacts	`<div ui-view="footer">`	contacts
footer@	`<div ui-view="footer">`	contacts
@contacts	`<div ui-view>`	contacts
@	`<div ui-view>`	contacts
sidebar@contacts.list	`<div ui-view="sidebar">`	contacts.list

$STATEPARAMS

$stateParams is an injectable object of URL params

```
// If you had a url on your state of:
url: '/users/:id/details/{type}/{repeat:[0-9]+}?from&to'

// Then you navigated your browser to:
'/users/123/details//0'

// Your $stateParams object would be
{ id:'123', type:'', repeat:'0' }

// Then you navigated your browser to:
'/users/123/details/default/0?from=there&to=here'

// Your $stateParams object would be
{ id:'123', type:'default', repeat:'0', from:'there', to:'here' }
```

IMPORTANT: $stateParams gives you access to the params for only the child state you are on, and NO parent state params.

```
app.controller('MyController', function ($scope, $stateParams) {
    $scope.params = $stateParams;
})
```

URL ROUTER

Much like the normal Angular Router, you can use:

- when() for redirection
- otherwise() for invalid routes
- rule() for custom url handling (takes a function)

```
app.config(function($urlRouterProvider){
    // When someone goes to admin, redirect to /index
    $urlRouterProvider.when('/admin', '/index');

    // Set a default route
    $urlRouterProvider.otherwise('/index');

    // What this function returns will be set as the $location.url
    $urlRouterProvider.rule(function ($injector, $location) {
        var path = $location.path(), normalized = path.toLowerCase();
        if (path != normalized) {
            // Replace current URL so we don't change states again.
            $location.replace().path(normalized);
        }
        // Because we've returned nothing, no state change occurs
    });
})
```

NAVIGATION

There are three main ways to activate a state:

- Call $state.go(). High-level convenience method.
- Normal "href" defined routing links using URLs
- State-based routing using "ui-sref"

```
$state.go("contacts.detail", {id: 2}, {inherit:false});
```

```
$state.transitionTo("contacts.detail");
```

```
<div>
    <a href="/contacts/detail">Contact Detail</a>
    <a ui-sref="contacts.list">Contacts List</a>
    <a ui-sref="contacts.detail({id: contact.id})">Contact Detail</a>
    <a ui-sref="contacts.detail" ui-sref-opts="{id: contact.id}">
        Contact Detail
    </a>
</div>
```

SWEET API

```
expect($state.href("about.person", { person: "bob" }))
  .toEqual("/about/bob");
```

```
expect($state.get())
  .toEqual([{
    name: "assets.details.topSection",
    url: "^/offers/:id/:topSection"
  }]);
```

```
$state.$current.name = 'contacts.details.item.url';

$state.includes("*.details.*.*"); // returns true
$state.includes("*.details.**"); // returns true
$state.includes("**.item.**"); // returns true
$state.includes("*.details.item.url"); // returns true
$state.includes("*.details.*.url"); // returns true
$state.includes("*.details.*"); // returns false
$state.includes("item.**"); // returns false
```

```
$state.is('contact.details.item'); // returns true
$state.is(contactDetailItemStateObject); // returns true
```

```
//$state.reload() is just an alias for:

$state.transitionTo($state.current, $stateParams, {
  reload: true, inherit: false, notify: false
});
```

HACKABLE!

What if I want a state change to slide in a side panel?

```
<div class="contacts" ng-controller="ContactsController" >
    <div ng-controller="ContactsListController"
         ng-class="{show: myCurrentState == listStateName}">
        <h1>Contacts</h1>
        <div ng-include="'/contact_list.html'"></div>
    </div>
    <div ng-controller="ContactDetailsController"
         ng-class="{show: myCurrentState == detailsStateName}">
        <div ng-include="'/contact_details.html'"></div>
    </div>

    <div ng-controller="ContactSidebarController"
         ng-class="{show: showSideBar}">
        <div ng-include="'/contact_sidebar.html'"></div>
    </div>
</div>
```

CONTACTS CONTROLLER

```
var ContactsController = function ($scope) {
    $scope.showSideBar = true;
    $scope.listStateName = 'contacts.list';
    $scope.detailsStateName = 'contacts.details';

    $scope.myCurrentState = $scope.listStateName;

    $scope.changeState = function (toState, toParams) {
        $scope.myCurrentState = toState.name;
        if (toState.name === 'contacts.list') {
            $scope.showSideBar = true;
        }
        else {
            $scope.showSideBar = false;
        }
    }

    var unListen = $scope.on('$stateChangeSuccess',
        function (event, toState, toParams, fromState, fromParams) {
            $scope.changeState(toState, toParams)
        }
    )
    $scope.on('$destroy', unListen)
}
```

RESOURCES

UI-Router Demo

API Docs

Wiki

THE END

Brought to you by reveal.js