AngularUI Router build passing (https://travis-ci.org/angular-ui/ui-router)

The de-facto solution to flexible routing with nested views

Download 0.2.17 (http://angular-ui.github.io/ui-router/release/angular-ui-router.js) (or **Minified (http://angular-ui.github.io/ui-router/release/angular-ui-router.min.js)**) |
Guide (https://github.com/angular-ui/ui-router/wiki) |
API (http://angular-ui.github.io/ui-router/site) |
Sample (http://angular-ui.github.com/ui-router/sample/) (**Src (https://github.com/angular-ui/ui-router/tree/gh-pages/sample)**) |
FAQ (https://github.com/angular-ui/ui-router/wiki/Frequently-Asked-Questions) |
Resources |
Report an Issue (https://github.com/angular-ui/ui-router/blob/master/CONTRIBUTING.md#report-an-issue) |
Contribute (https://github.com/angular-ui/ui-router/blob/master/CONTRIBUTING.md#contribute) |
Help! (http://stackoverflow.com/questions/ask?tags=angularjs,angular-ui-router) |
Discuss (https://groups.google.com/forum/#!categories/angular-ui/router)

AngularUI Router is a routing framework for AngularJS (http://angularjs.org), which allows you to organize the
parts of your interface into a *state machine* (https://en.wikipedia.org/wiki/Finite-state_machine). Unlike the
`$route` service (http://docs.angularjs.org/api/ngRoute.$route) in the Angular ngRoute module, which is organized around URL
routes, UI-Router is organized around *states* (https://github.com/angular-ui/ui-router/wiki), which may optionally have routes, as well as other behavior, attached.

States are bound to *named*, *nested* and *parallel views*, allowing you to powerfully manage your application's interface.

Check out the sample app: http://angular-ui.github.io/ui-router/sample/

-

Note: *UI-Router is under active development. As such, while this library is well-tested, the API may change. Consider using it in production applications only if you're comfortable following a changelog and updating your usage accordingly.*

Get Started

(1) Get UI-Router in one of the following ways:

- clone & build (CONTRIBUTING.md#developing) this repository
- download the release (http://angular-ui.github.io/ui-router/release/angular-ui-router.js) (or minified (http://angular-ui.github.io/ui-router/release/angular-ui-router.min.js))
- link to cdn (http://cdnjs.com/libraries/angular-ui-router)

- via **[jspm (http://jspm.io/)](http://jspm.io/)**: by running `$ jspm install angular-ui-router` from your console
- or via **[npm (https://www.npmjs.org/)](https://www.npmjs.org/)**: by running `$ npm install angular-ui-router` from your console
- or via **[Bower (http://bower.io/)](http://bower.io/)**: by running `$ bower install angular-ui-router` from your console
- or via **[Component (https://github.com/component/component)](https://github.com/component/component)**: by running `$ component install angular-ui/ui-router` from your console

(2) Include `angular-ui-router.js` (or `angular-ui-router.min.js`) in your `index.html`, after including Angular itself (For Component users: ignore this step)

(3) Add `'ui.router'` to your main module's list of dependencies (For Component users: replace `'ui.router'` with `require('angular-ui-router')`)

When you're done, your setup should look similar to the following:

```
<!doctype html>
<html ng-app="myApp">
<head>
    <script
src="//ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
    <script src="js/angular-ui-router.min.js"></script>
    <script>
        var myApp = angular.module('myApp', ['ui.router']);
        // For Component users, it should look like this:
        // var myApp = angular.module('myApp', [require('angular-ui-router')]);
    </script>
    ...
</head>
<body>
    ...
</body>
</html>
```

[Nested States & Views (http://plnkr.co/edit/u18KQc?p=preview)](http://plnkr.co/edit/u18KQc?p=preview)

The majority of UI-Router's power is in its ability to nest states & views.

(1) First, follow the setup instructions detailed above.

(2) Then, add a [`ui-view` directive (https://github.com/angular-ui/ui-router/wiki/Quick-Reference#ui-view)](https://github.com/angular-ui/ui-router/wiki/Quick-Reference#ui-view) to the `<body />` of your app.

```
<!-- index.html -->
<body>
    <div ui-view></div>
    <!-- We'll also add some navigation: -->
    <a ui-sref="state1">State 1</a>
    <a ui-sref="state2">State 2</a>
</body>
```

(3) You'll notice we also added some links with `ui-sref` directives (https://github.com/angular-ui/ui-router/wiki/Quick-Reference#ui-sref). In addition to managing state transitions, this directive auto-generates the `href` attribute of the `<a />` element it's attached to, if the corresponding state has a URL. Next we'll add some templates. These will plug into the `ui-view` within `index.html`. Notice that they have their own `ui-view` as well! That is the key to nesting states and views.

```html
<!-- partials/state1.html -->
<h1>State 1</h1>
<hr/>
<a ui-sref="state1.list">Show List</a>
<div ui-view></div>
```

```html
<!-- partials/state2.html -->
<h1>State 2</h1>
<hr/>
<a ui-sref="state2.list">Show List</a>
<div ui-view></div>
```

(4) Next, we'll add some child templates. *These* will get plugged into the `ui-view` of their parent state templates.

```html
<!-- partials/state1.list.html -->
<h3>List of State 1 Items</h3>
<ul>
  <li ng-repeat="item in items">{{ item }}</li>
</ul>
```

```html
<!-- partials/state2.list.html -->
<h3>List of State 2 Things</h3>
<ul>
  <li ng-repeat="thing in things">{{ thing }}</li>
</ul>
```

(5) Finally, we'll wire it all up with `$stateProvider`. Set up your states in the module config, as in the following:

```
myApp.config(function($stateProvider, $urlRouterProvider) {
  //
  // For any unmatched url, redirect to /state1
  $urlRouterProvider.otherwise("/state1");
  //
  // Now set up the states
  $stateProvider
    .state('state1', {
      url: "/state1",
      templateUrl: "partials/state1.html"
    })
    .state('state1.list', {
      url: "/list",
      templateUrl: "partials/state1.list.html",
      controller: function($scope) {
        $scope.items = ["A", "List", "Of", "Items"];
      }
    })
    .state('state2', {
      url: "/state2",
      templateUrl: "partials/state2.html"
    })
    .state('state2.list', {
      url: "/list",
      templateUrl: "partials/state2.list.html",
      controller: function($scope) {
        $scope.things = ["A", "Set", "Of", "Things"];
      }
    });
});
```

(6) See this quick start example in action.

Go to Quick Start Plunker for Nested States & Views (http://plnkr.co/edit/u18KQc?p=preview)

(7) This only scratches the surface

Dive Deeper! (https://github.com/angular-ui/ui-router/wiki)

Multiple & Named Views (http://plnkr.co/edit/SDOcGS?p=preview)

Another great feature is the ability to have multiple `ui-views` view per template.

Pro Tip: *While multiple parallel views are a powerful feature, you'll often be able to manage your interfaces more effectively by nesting your views, and pairing those views with nested states.*

(1) Follow the setup instructions detailed above.

(2) Add one or more `ui-view` to your app, give them names.

```
<!-- index.html -->
<body>
    <div ui-view="viewA"></div>
    <div ui-view="viewB"></div>
    <!-- Also a way to navigate -->
    <a ui-sref="route1">Route 1</a>
    <a ui-sref="route2">Route 2</a>
</body>
```

(3) Set up your states in the module config:

```
myApp.config(function($stateProvider) {
  $stateProvider
    .state('index', {
      url: "",
      views: {
        "viewA": { template: "index.viewA" },
        "viewB": { template: "index.viewB" }
      }
    })
    .state('route1', {
      url: "/route1",
      views: {
        "viewA": { template: "route1.viewA" },
        "viewB": { template: "route1.viewB" }
      }
    })
    .state('route2', {
      url: "/route2",
      views: {
        "viewA": { template: "route2.viewA" },
        "viewB": { template: "route2.viewB" }
      }
    })
});
```

(4) See this quick start example in action.

Go to Quick Start Plunker for Multiple & Named Views (http://plnkr.co/edit/SDOcGS?p=preview)

Resources

- In-Depth Guide (https://github.com/angular-ui/ui-router/wiki)
- API Reference (http://angular-ui.github.io/ui-router/site)
- Sample App (http://angular-ui.github.com/ui-router/sample/) (Source (https://github.com/angular-ui/ui-router/tree/gh-pages/sample))

- FAQ (https://github.com/angular-ui/ui-router/wiki/Frequently-Asked-Questions)
- Slides comparing ngRoute to ui-router (http://slid.es/timkindberg/ui-router#/)
- UI-Router Extras / Addons (http://christopherthielen.github.io/ui-router-extras/#/home) (@christopherthielen)

Videos

- Introduction Video (https://egghead.io/lessons/angularjs-introduction-ui-router) (egghead.io)
- Tim Kindberg on Angular UI-Router (https://www.youtube.com/watch?v=lBqiZSemrqg)
- Activating States (https://egghead.io/lessons/angularjs-ui-router-activating-states) (egghead.io)
- Learn Angular.js using UI-Router (http://youtu.be/QETUuZ27N0w) (LearnCode.academy)

Reporting issues and Contributing

Please read our Contributor guidelines (CONTRIBUTING.md) before reporting an issue or creating a pull request.