/ Tutorial / 10 - Event Handlers

In this step, you will add a clickable phone image swapper to the phone details page.

- The phone details view displays one large image of the current phone and several smaller thumbnail images. It would be great if we could replace the large image with any of the thumbnails just by clicking on the desired thumbnail image. Let's have a look at how we can do this with Angular.

Workspace Reset Instructions ➤

The most important changes are listed below. You can see the full diff on GitHub

Controller

`app/js/controllers.js` :

```
...
var phonecatControllers = angular.module('phonecatControllers',[]);

phonecatControllers.controller('PhoneDetailCtrl', ['$scope', '$routeParams', '$http',
  function($scope, $routeParams, $http) {
    $http.get('phones/' + $routeParams.phoneId + '.json').success(function(data) {
      $scope.phone = data;
      $scope.mainImageUrl = data.images[0];
    });

    $scope.setImage = function(imageUrl) {
      $scope.mainImageUrl = imageUrl;
    }
  }]);
```

In the `PhoneDetailCtrl` controller, we created the `mainImageUrl` model property and set its default value to the first phone image URL.

We also created a `setImage` event handler function that will change the value of `mainImageUrl`.

Template

app/partials/phone-detail.html :

```
<img ng-src="{{mainImageUrl}}" class="phone">

...

<ul class="phone-thumbs">
  <li ng-repeat="img in phone.images">
    <img ng-src="{{img}}" ng-click="setImage(img)">
  </li>
</ul>
...
```

We bound the `ngSrc` directive of the large image to the `mainImageUrl` property.

We also registered an `ngClick` handler with thumbnail images. When a user clicks on one of the thumbnail images, the handler will use the `setImage` event handler function to change the value of the `mainImageUrl` property to the URL of the thumbnail image.

Test

To verify this new feature, we added two end-to-end tests. One verifies that the main image is set to the first phone image by default. The second test clicks on several thumbnail images and verifies that the main image changed appropriately.

test/e2e/scenarios.js :

```
...
describe('Phone detail view', function() {

...

  it('should display the first phone image as the main phone image', function() {
    expect(element(by.css('img.phone')).getAttribute('src')).toMatch(/img\/phones\/nexus-s.0.jpg/);
  });


  it('should swap main image if a thumbnail image is clicked on', function() {
    element(by.css('.phone-thumbs li:nth-child(3) img')).click();
    expect(element(by.css('img.phone')).getAttribute('src')).toMatch(/img\/phones\/nexus-s.2.jpg/);

    element(by.css('.phone-thumbs li:nth-child(1) img')).click();
    expect(element(by.css('img.phone')).getAttribute('src')).toMatch(/img\/phones\/nexus-s.0.jpg/);
  });
});
```

You can now rerun `npm run protractor` to see the tests run.

You also have to refactor one of your unit tests because of the addition of the `mainImageUrl` model property to the `PhoneDetailCtrl` controller. Below, we create the function `xyzPhoneData` which returns the appropriate json with the `images` attribute in order to get the test to pass.

test/unit/controllersSpec.js :

```
...
 beforeEach(module('phonecatApp'));

...

describe('PhoneDetailCtrl', function(){
   var scope, $httpBackend, ctrl,
       xyzPhoneData = function() {
         return {
           name: 'phone xyz',
           images: ['image/url1.png', 'image/url2.png']
         }
       };


   beforeEach(inject(function(_$httpBackend_, $rootScope, $routeParams, $controller) {
     $httpBackend = _$httpBackend_;
     $httpBackend.expectGET('phones/xyz.json').respond(xyzPhoneData());

     $routeParams.phoneId = 'xyz';
     scope = $rootScope.$new();
     ctrl = $controller('PhoneDetailCtrl', {$scope: scope});
   }));


   it('should fetch phone detail', function() {
     expect(scope.phone).toBeUndefined();
     $httpBackend.flush();

     expect(scope.phone).toEqual(xyzPhoneData());
   });
 });
```

Your unit tests should now be passing.

Experiments

Let's add a new controller method to `PhoneDetailCtrl`:

```
$scope.hello = function(name) {
    alert('Hello ' + (name || 'world') + '!');
}
```

and add:

```
<button ng-click="hello('Elmo')">Hello</button>
```

to the `phone-detail.html` template.

Summary

With the phone image swapper in place, we're ready for step 11 to learn an even better way to fetch data.

Previous Live Demo Code Diff Next

Super-powered by Google ©2010-2014 (v1.3.0 superluminal-nudge) Back to top

Code licensed under the The MIT License. Documentation licensed under CC BY 3.0.