v1.5.6-build.4831 (snapshot

/ Tutorial (tutorial)
/ 10 - More Templating (tutorial/step_10)

ocs/content/tutorial/step_10.ngdoc?message=docs(tutorial%2F10 - More Templating)%3A%20describe%20your%20change...)

Tutorial (tutorial)

0 - Bootstrapping (tutorial/step_00)
1 - Static Template (tutorial/step_01)
2 - Angular Templates (tutorial/step_02)
3 - Components (tutorial/step_03)
4 - Directory and File Organization (tutorial/step_04)
5 - Filtering Repeaters (tutorial/step_05)
6 - Two-way Data Binding (tutorial/step_06)
7 - XHR & Dependency Injection (tutorial/step_07)
8 - Templating Links & Images (tutorial/step_08)
9 - Routing & Multiple Views (tutorial/step_09)
10 - More Templating (tutorial/step_10)
11 - Custom Filters (tutorial/step_11)
12 - Event Handlers (tutorial/step_12)
13 - REST and Custom Services (tutorial/step_13)
14 - Animations (tutorial/step_14)
The End (tutorial/the_end)

Previous (tutorial/step_09) **▶ Live Demo** (http://angular.github.io/angular-phonecat/step-10/app) **Code Diff** (https://github.com/angular/angular-phonecat/compare/step-9...step-10) **Next** (tutorial/step_11)

In this step, we will implement the phone details view, which is displayed when a user clicks on a phone in the phone list.

- When you click on a phone on the list, the phone details page with phone-specific information is displayed.

To implement the phone details view we are going to use $http (api/ng/service/$http) to fetch our data, and then flesh out the `phoneDetail` component's template.

Workspace Reset Instructions ➤

The most important changes are listed below. You can see the full diff on GitHub (https://github.com/angular/angular-phonecat/compare/step-9...step-10).

Data

In addition to `phones.json`, the `app/phones/` directory also contains one JSON file for each phone:

`app/phones/nexus-s.json`: (sample snippet)

```
{
  "additionalFeatures": "Contour Display, Near Field
Communications (NFC), ...",
  "android": {
    "os": "Android 2.3",
    "ui": "Android"
  },
  ...
  "images": [
    "img/phones/nexus-s.0.jpg",
    "img/phones/nexus-s.1.jpg",
    "img/phones/nexus-s.2.jpg",
    "img/phones/nexus-s.3.jpg"
  ],
  "storage": {
    "flash": "16384MB",
    "ram": "512MB"
  }
}
```

Each of these files describes various properties of the phone using the same data structure. We will show this data in the phone details view.

Component Controller

We will expand the `phoneDetail` component's controller by using the `$http` service to fetch the appropriate JSON files. This works the same way as the `phoneList` component's controller.

app/phone-detail/phone-detail.component.js :

```
angular.
  module('phoneDetail').
  component('phoneDetail', {
    templateUrl: 'phone-detail/phone-detail.template.html',
    controller: ['$http', '$routeParams',
      function PhoneDetailController($http, $routeParams) {
        var self = this;

        $http.get('phones/' + $routeParams.phoneId +
'.json').then(function(response) {
          self.phone = response.data;
        });
      }
    ]
  });
```

To construct the URL for the HTTP request, we use `$routeParams.phoneId` , which is extracted from the current route by the `$route` service.

Component Template

The inline, TBD placeholder template has been replaced with a full blown external template, including lists and bindings that comprise the phone details. Note how we use the Angular `{{expression}}` markup and `ngRepeat` to project phone data from our model into the view.

app/phone-detail/phone-detail.template.html：

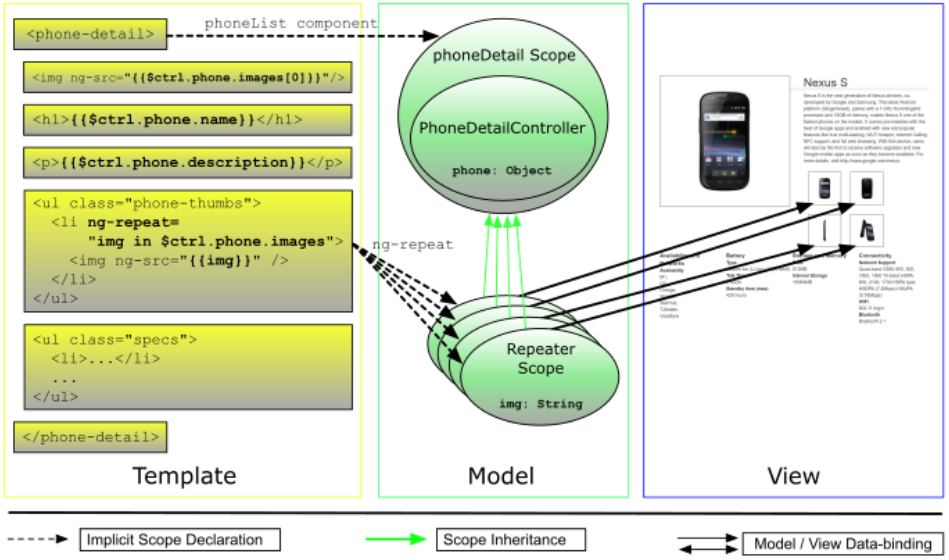
```
<img ng-src="{{$ctrl.phone.images[0]}}" class="phone" />

<h1>{{$ctrl.phone.name}}</h1>

<p>{{$ctrl.phone.description}}</p>

<ul class="phone-thumbs">
  <li ng-repeat="img in $ctrl.phone.images">
    <img ng-src="{{img}}" />
  </li>
</ul>

<ul class="specs">
  <li>
    <span>Availability and Networks</span>
    <dl>
      <dt>Availability</dt>
      <dd ng-repeat="availability in
$ctrl.phone.availability">{{availability}}</dd>
    </dl>
  </li>
  ...
  <li>
    <span>Additional Features</span>
    <dd>{{$ctrl.phone.additionalFeatures}}</dd>
  </li>
</ul>
```



Testing

We wrote a new unit test that is similar to the one we wrote for the `phoneList` component's controller in step 7 (tutorial/step_07#testing).

`app/phone-detail/phone-detail.component.spec.js`:

```
describe('phoneDetail', function() {

  // Load the module that contains the `phoneDetail`
component before each test
  beforeEach(module('phoneDetail'));

  // Test the controller
  describe('PhoneDetailController', function() {
    var $httpBackend, ctrl;

    beforeEach(inject(function($componentController,
_$httpBackend_, $routeParams) {
      $httpBackend = _$httpBackend_;

$httpBackend.expectGET('phones/xyz.json').respond({name:
'phone xyz'});

      $routeParams.phoneId = 'xyz';

      ctrl = $componentController('phoneDetail');
    }));

    it('should fetch the phone details', function() {
      expect(ctrl.phone).toBeUndefined();

      $httpBackend.flush();
      expect(ctrl.phone).toEqual({name: 'phone xyz'});
    });

  });

});
```

You should now see the following output in the Karma tab:

```
Chrome 49.0: Executed 3 of 3 SUCCESS (0.159 secs / 0.136
secs)
```

We also added a new E2E test that navigates to the 'Nexus S' details page and verifies that the heading on the page is "Nexus S".

e2e-tests/scenarios.js

```
...

describe('View: Phone detail', function() {

  beforeEach(function() {
    browser.get('index.html#!/phones/nexus-s');
  });

  it('should display the `nexus-s` page', function() {

expect(element(by.binding('$ctrl.phone.name')).getText()).to
Be('Nexus S');
  });

});

...
```

You can run the tests with `npm run protractor`.

Experiments

- Using Protractor's API (https://angular.github.io/protractor/#/api), write a test that verifies that we display 4 thumbnail images on the 'Nexus S' details page.

Summary

Now that the phone details view is in place, proceed to step 11 (tutorial/step_11) to learn how to write your own custom display filter.

Previous (tutorial/step_09) ▶ Live Demo (http://angular.github.io/angular-phonecat/step-10/app)

Code Diff (https://github.com/angular/angular-