## Testing Routing

**Learning Objectives**

* Understand how to configure the ATB in order to test routing.
* Understand how to write test specs that involve routing.

**Test setup**

To test routing we need a few components and a route configuration:

import {Component} from "@angular/core";

import {Routes} from "@angular/router";

@Component({

template: `Search`

})

export class SearchComponent {

}

@Component({

template: `Home`

})

export class HomeComponent {

}

@Component({

template: `<router-outlet></router-outlet>`

})

export class AppComponent {

}

export const routes: Routes = [

{path: '', redirectTo: 'home', pathMatch: 'full'},

{path: 'home', component: HomeComponent},

{path: 'search', component: SearchComponent}

]

;

We create three components HomeComponent, SearchComponent and an AppComponent with a <routeroutlet>.

We also create a route configuration where '' redirects you to home and home and search show their respective components.

Our basic test suite looks like so:

import {Location} from "@angular/common";

import {TestBed, fakeAsync, tick} from '@angular/core/testing';

import {RouterTestingModule} from "@angular/router/testing";

import {Router} from "@angular/router";

import {

HomeComponent,

SearchComponent,

AppComponent,

routes

}

from "./router"

describe('Router: App', () => {

beforeEach(() => {

TestBed.configureTestingModule({

declarations: [

①

HomeComponent,

SearchComponent,

AppComponent

]

});

});

})

;

① We import and declare our components in the test bed configuration.

**Router setup**

Normally to setup routing in an Angular application we import the RouterModule and *provide* the routes to the NgModule with RouterModule.withRoutes(routes).

However when testing routing we use the RouterTestingModule instead. This modules sets up the router with a *spy* implementation of the *Location Strategy* that doesn’t actually change the URL.

We also need to get the injected Router and Location so we can use them in the test specs.

Our test suite file now looks like:

describe('Router: App', () => {

let location: Location;

let router: Router;

let fixture;

beforeEach(() => {

TestBed.configureTestingModule({

imports: [RouterTestingModule.withRoutes(routes)],

①

declarations: [

HomeComponent,

SearchComponent,

AppComponent

]

});

router = TestBed.get(Router);

②

location = TestBed.get(Location);

③

fixture = TestBed.createComponent(AppComponent);

④

router.initialNavigation();

⑤

});

})

;

1. We import our RouterTestingModule with our routes.
2. We grab a reference to the injected Router.
3. We grab a reference to the injected Location.
4. We ask the test bed to create an instance of our root AppComponent. We don’t need this reference in our test specs but we do need to create the root component with the router-outlet so the router has somewhere to insert components.
5. This sets up the location change listener and performs the initial navigation.

We are now ready to create our test specs.

**Testing routing**

In our configuration we’ve set it so if you land on the root *empty* url you will be redirected to /home, lets add a test spec for this:

it('navigate to "" redirects you to /home', fakeAsync(() => {

①

router.navigate(['']);

②

tick();

③

expect(location.path()).toBe('/home');

④

}))

;

1. Routing is an asynchronous activity so we use one of the asynchronous testing methods at our

disposal, in this case the fakeAsync method.

1. We trigger the router to navigate to the empty path.
2. We wait for all pending promises to be resolved.
3. We can then inspect the *path* our application should be at with location.path() Lets also add a test spec for navigating to the search route, like so:

it('navigate to "search" takes you to /search', fakeAsync(() => {

router.navigate(['search']);

tick();

expect(location.path()).toBe('/search');

}))

;

The spec is exactly the same as the previous one, our link params array is different since we are triggering a different route and our expectation is again different, but the rest is the same.

**Summary**

We can test routing in Angular by using RouterTestingModule instead of RouterModule to provide our routes.

This uses a *spy* implementation of Location which doesn’t trigger a request for a new URL but does let us know the target URL which we can use in our test specs.

**Listing**

*router.ts*

import {Component} from "@angular/core";

import {Routes} from "@angular/router";

@Component({

template: `Search`

})

export class SearchComponent {

}

@Component({

template: `Home`

})

export class HomeComponent {

}

@Component({

template: `<router-outlet></router-outlet>`

})

export class AppComponent {

}

export const routes: Routes = [

// {path: '', redirectTo: 'home', pathMatch: 'full'},

{path: 'home', component: HomeComponent},

{path: 'search', component: SearchComponent}

]

;

*router.spec.ts*

/\* tslint:disable:no-unused-variable \*/

import {Location} from "@angular/common";

import {TestBed, fakeAsync, tick} from '@angular/core/testing';

import {RouterTestingModule} from "@angular/router/testing";

import {Router} from "@angular/router";

import {

HomeComponent,

SearchComponent,

AppComponent,

routes

}

from "./router"

describe('Router: App', () => {

let location: Location;

let router: Router;

let fixture;

beforeEach(() => {

TestBed.configureTestingModule({

imports: [ RouterTestingModule.withRoutes(routes)],

declarations: [

HomeComponent,

SearchComponent,

AppComponent

]

});

router = TestBed.get(Router);

location = TestBed.get(Location);

fixture = TestBed.createComponent(AppComponent);

router.initialNavigation();

});

it('fakeAsync works', fakeAsync(() => {

let promise = new Promise((resolve) => {

setTimeout(resolve, 10)

});

let done = false;

promise.then(() => done = true);

tick(50);

expect(done).toBeTruthy();

}));

it('navigate to "" redirects you to /home', fakeAsync(() => {

router.navigate(['']);

tick(50);

expect(location.path()).toBe('/home');

}));

it('navigate to "search" takes you to /search', fakeAsync(() => {

router.navigate(['/search']);

tick(50);

expect(location.path()).toBe('/search');

}));

})

;