Angular Lisbon Meetup 17.4.2018

Testing with Angular

Peter Bouda, hey@peterbouda.eu

ng-lisbon.com

About Me

- Web Developer since the 90s
- LoopBack + Angular since 2015
- Senior Web Architect at Apiax
- Angular trainer at ng-lisbon.com
- The rest is LEGO
- https://www.peterbouda.eu/

Why Testing?

- Protect against regressions
- Describe functionality of units
- Make weaknesses visible in design and architecture of the app
    - If something is hard to test then probably you can improve the design

Jasmine, Karma and Angular

- Jasmine is a testing framework and delivers the syntax for Angular testing
- Karma is a test runner and gives you feedback about the outcome of the tests
- Angular Testing Utilities are Angular specific helpers to test your code in interaction with Angular
    - Often services and pipes can be tested in complete isolation!

Angular Testing Utilities

- `TestBed` to generate a special, module-like test environment
- Within the `TestBed` we can instantiate and test the unit
- The `TestBed` gives us access to a `fixture` with a `DebugElement`
    - Fixture: the test environment plus the component to test (e.g. a components)
    - DebugElement: access the DOM of a unit

Jasmine Syntax

```
describe('ProductComponent', () => {
  beforeEach(() => {
    // Initialize
  });
  it('should create', () => {
    expect(component).toBeTruthy();
  });
  it('should have a the productId initalized', () => {
    expect(component.productId).toBe(0);
  });
});
```

Testing a component

```
import { async, ComponentFixture, TestBed } from '@angular/core/testing';
import { By } from '@angular/platform-browser';
import { InfoBoxComponent } from './info-box.component';

describe('InfoBoxComponent', () => {
  let component: InfoBoxComponent;
  let fixture: ComponentFixture<InfoBoxComponent>;
  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [ InfoBoxComponent ]
    })
    .compileComponents();
  }));
  beforeEach(() => {
    fixture = TestBed.createComponent(InfoBoxComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });
  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Change Detection

- The `TestBed` does not perform automatic change detection
- You can trigger a change detection run with fixture.detectChanges()
- Or automatic:

```
import { ComponentFixtureAutoDetect } from '@angular/core/testing';
TestBed.configureTestingModule({
  declarations: [ BannerComponent ],
  providers: [
    { provide: ComponentFixtureAutoDetect, useValue: true }
  ]
});
```

Accessing the View

- The `fixture` has two properties `debugElement` and `nativeElement`
- `debugElement` contains helper methods, e.g.: `fixture.debugElement.queryAll(By.css('button'));`
- `nativeElement` allows direct access to the DOM, e.g.: `fixture.nativeElement.click()`

Example: Testing a component

```
it('should display the set title', () => {
  component.title = 'Test Title';
  let de = fixture.debugElement.query(By.css('h1'));
  let el = de.nativeElement;
  fixture.detectChanges();
  expect(el.textContent).toContain('Test Title');
});
```

Best Pratice: Page Object

```
import { ComponentFixture } from '@angular/core/testing';
import { By } from '@angular/platform-browser';
import { OrganizationsHomeComponent } from './home.component';
export class Page {
  get searchInput() {
    return this.query('input[placeholder="Search Companies"]');
  }
  get createCompanyButton() {
    return this.query('header a');
  }
  public searchSpy: jasmine.Spy;
  constructor(private fixture: ComponentFixture<OrganizationsHomeComponent>) {
    const component = fixture.componentInstance;
    this.searchSpy = spyOn(component, 'search');
  }
  private query(selector: string) {
    return this.fixture.debugElement.query(By.css(selector));
  }
  private queryAll(selector: string) {
    return this.fixture.debugElement.queryAll(By.css(selector));
  }
}
```

Dependencies

- Goals is to test units without dependencies
- Mock/Stub to replace dependencies, e.g. services
- Capture method calls and replace methods with spies:

```
spyOn(component,'onSearch').and.callFake(() => { ... })
```

ng-lisbon.com

Component Stubs

```
@Component({
  template: '',
  selector: 'router-outlet'
})
class MockRouterOutlet {}
...
beforeEach(async(() => {
  TestBed.configureTestingModule({
    declarations: [
      AppComponent,
      MockRouterOutlet
    ],
  });
  TestBed.compileComponents();
}));
```

Service Stubs

```
class MockProductService {}
...
beforeEach(async(() => {
  TestBed.configureTestingModule({
    imports: [ FormsModule ],
    declarations: [ ProductDetailComponent ],
    providers: [
      { provide: ProductService, useClass: MockProductService }
    ]
  })
  .compileComponents();
}));
```

Using Spies

```
const spySubmit = spyOn(component, 'onSubmit').and.callThrough();
expect(spySubmit.calls.any()).toBe(false);
expect(spySubmit.calls.count()).toBe(2);

const prodService = fixture.debugElement.injector.get(ProductService);
spyOn(prodService, 'getProduct').and.callFake(
  (prodId) => return new Product('Titel', 'Beschreibung')
);
```

Testing Inputs

```
it('should display product title', () => {
  const testProduct = new Product("Titel", "Beschreibung");
  const prodElem = fixture.debugElement.query(By.css('.product'));
  component.product = testProduct;
  fixture.detectChanges();
  expect(prodElem.nativeElement.textContent).toContain(testProduct.title);
});
```

Testing Outputs

```
it('should output a submitted event when save button is clicked', () => {
  let submitted = false;
  const btnSubmit = fixture.debugElement.query(By.css('button')).nativeElement;
  component.submitted.subscribe(() => submitted = true);
  btnSubmit.click();
  expect(submitted).toBe(true);
});
```

Testing Services

```
import { ProductService } from './product.service';
import { Product } from '../shared/product.model';

describe('ProductService', () => {
  let productService: ProductService;
  beforeEach(() => {
    productService = new ProductService();
  });
  it('should return an empty list initially', () => {
    expect(productService.getProducts()).toEqual([]);
  });
  it('should return an the entry after addProduct', () => {
    const testProduct = new Product('Titel', 'Beschreibung');
    productService.addProduct(testProduct);
    expect(productService.getProducts()).toEqual([testProduct]);
  });
});
```

Testing Directives

```
@Component({
  template: '<div myDirective></div>'
})
class TestComponent {};
// ... and then as in the component test
beforeEach(async(() => {
  TestBed.configureTestingModule({
    declarations: [ TestComponent, MyDirective ]
  })
  .compileComponents();
}));
it('should change background color when clicked', () => {
  const element = fixture.debugElement.query(By.directive(MyDirective));
  // const highlight = element.injector.get(MyDirective) as MyDirective;
  element.debugElement.click();
  expect(element.nativeElement.style.backgroundColor).toBe('blue');
});
```

Thanks!

Peter Bouda, hey@peterbouda.eu

ng-lisbon.com