Unit test directive :

Let us test an directive . The directive is about changing the color of text when mouse is over/out.For this I am creating a directive with name **color.directive.ts**.

import {Directive,HostListener,HostBinding} from '@angular/core';

@Directive({selector : '[myColor]'})

export class MyColor{

private col : string = 'pink';

@HostListener('mouseover') mouseOver(){

console.log('mouse over')

this.col = 'red';

}

@HostListener('mouseout') mouseOut(){

console.log('mouse out')

this.col = 'green';

}

@HostBinding('style.color') get color(){

return this.col;

}

}

For testing the directive we need to also create an component so that we can test it on proxy environment. So , I hahve created an component with name **gross.component.ts.**

import { Component, OnInit } from '@angular/core';

@Component({

selector: 'app-gross',

templateUrl: './gross.component.html',

styleUrls: ['./gross.component.css']

})

export class GrossComponent implements OnInit {

constructor() { }

ngOnInit() {

}

}

Now, let’s write the spec file with name **gross.component.spec.ts**

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

import { async, ComponentFixture, TestBed } from '@angular/core/testing';

import { By } from '@angular/platform-browser';

import { DebugElement } from '@angular/core';

import { GrossComponent } from './gross.component';

import {MyColor} from '../directives/color.directive';

describe('GrossComponent', () => {

let component: GrossComponent;

let fixture: ComponentFixture<GrossComponent>;

beforeEach(async(() => {

TestBed.configureTestingModule({

declarations: [ GrossComponent,MyColor ]

})

.compileComponents();

}));

beforeEach(() => {

fixture = TestBed.createComponent(GrossComponent);

component = fixture.componentInstance;

fixture.detectChanges();

});

it('should create', () => {

expect(component).toBeTruthy();

});

**it('Test for the mouserover event',()=>{**

**var ele = fixture.debugElement.query(By.css('span'))**

**ele.triggerEventHandler('mouseover',null);**

**fixture.detectChanges();**

**expect(ele.nativeElement.style.color).toBe('red')**

**});**

});

Let us now , discuss on above line marked :

Here **ele** contains the reference for span. We are using query() method for this. After then we are triggering the mouseover event on it . Just after detecting the changes , we are applying our own asserting so that when mouse gets over the span the color will be changes by red.

So, our test for the directive gets passed . We can also apply the sdame test for mouseout.