PROGRAMACIÓN DE APLICACIONES MÓVILES

PGY4121

Profesor:

DuocUC®

ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Actividad N°3.1: Testeando Nuestra Aplicación

Testing Ionic Angular

DuocUC®

Objetivos

Lo que se espera que aprendas en esta Actividad es:

- Unit Test
- Testeando con Jasmine

DuocUC®

¿Qué es Native APIs?

DuocUC®

PGY4121 Introducción

- El objetivo de las pruebas no es verificar que el código sea correcto, sino encontrar problemas dentro del código. Esta es una distinción sutil pero importante.

- Si nos propusimos demostrar que el código es correcto, es más probable que sigamos el camino feliz a través del código. Si nos propusimos encontrar problemas, es más probable que ejercitemos más el código y encontremos los errores que acechan allí

https://ionicframework.com/docs/angular/testing

DuocUC®

Pruebas Unitarias

DuocUC®

PGY4121 Claves

- Unidad de codigo: Elemento a probar, que pueden ser una clase o función
- Mock Object: Objeto simulado con que se ejecuta un test de forma aislada
- Inyección: Delegar a otro la responsabilidad de la creación de instancias de un componente
- Describe(): Descripción de la unidad de código
- It(): Característica a probar
- expect(): Mock Object a probar
- inject(): Test mediante Inyección

DuocUC®

PGY4121 Pruebas Unitarias

- ¿Dónde se aplica?
  - Se aplica a una unidad de código tales como Component, Page, Service o Pipes entre otros de forma aislada al resto del sistema.
- ¿Cómo se aísla?
  - Se inyectan objetos simulados en lugar de dependencias de código.
- ¿Que es un objeto simulado?
  - son objetos que imitan el comportamiento de objetos reales de forma controlada

DuocUC®

Jasmine

PGY4121 Usando Mocks con Jasmine

- Para hacer pruebas en ionic, la documentación oficial recomienda Jasmine, esta crea objetos simulados llamados espías por Jasmine que toma lugar de las dependencias durante las pruebas.

- Si modificamos la dependencia el Mock Object puede controlar los valores devueltos a esa dependencia.

DuocUC®

PGY4121 Usando Mocks con Jasmine

- Creación de Spies:
  - Desde 0 usando: jasmine.createSpy o jasmine.createSpyObj
  - Desde objetos existentes: spyOn() o spyOnProperty()

PGY4121 Usando Mocks con Jasmine

- jasmine.createSpy o jasmine.createSpyObj:
  - Se puede crear objetos simulados completos con conjuntos de métodos definidos en la creación
  - Es simple
  - Pero el objeto creado puede no coincidir con objetos reales

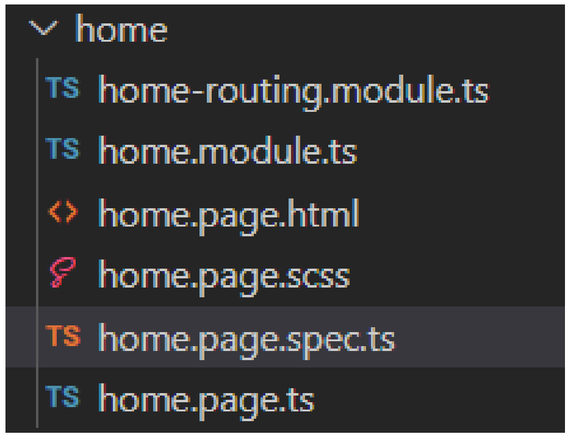
PGY4121 Usando Mocks con Jasmine

- spyOn() o spyOnProperty():
  - Se crea un spy de un objeto existente
  - Si se espía un método que no existe generará una excepción.
  - El SpyOnProperty espía una propiedad pero no un método.

Estructura y Funcionamiento General

PGY4121 Estructura General

- Al generar un Component, Page, Service, Pipe etc, se genera un archivo spec como por ejemplo home.page.ts se crea un archivo home.page.spec.ts



PGY4121 Estructura General

- Archivo spec.ts generado automáticamente de un Page

```
import { HomePage } from './home.page';

describe('HomePage', () => {
  let component: HomePage;
  let fixture: ComponentFixture<HomePage>;

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [ HomePage ],
      imports: [IonicModule.forRoot()]
    }).compileComponents();

    fixture = TestBed.createComponent(HomePage);
    component = fixture.componentInstance;
    fixture.detectChanges();
  }));

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

PGY4121 Estructura General

- Definición de la prueba general llamada ‘HomePage’ que contiene descripciones anidadas que definen principales áreas de funcionalidad o unidades de código

```
5
6  describe('HomePage', () => {
7    let component: HomePage;
8    let fixture: ComponentFixture<HomePage>;
9
10   beforeEach(async(() => {
11     TestBed.configureTestingModule({
12       declarations: [ HomePage ],
13       imports: [IonicModule.forRoot()]
14     }).compileComponents();
15
16     fixture = TestBed.createComponent(HomePage);
17     component = fixture.componentInstance;
18     fixture.detectChanges();
19   }));
20
21   it('should create', () => {
22     expect(component).toBeTruthy();
23   });
24 });
```

PGY4121 Estructura General

- Código de configuración

```
5
6   describe('HomePage', () => {
7     let component: HomePage;
8     let fixture: ComponentFixture<HomePage>;
9
10    beforeEach(async(() => {
11      TestBed.configureTestingModule({
12        declarations: [ HomePage ],
13        imports: [IonicModule.forRoot()]
14      }).compileComponents();
15
16      fixture = TestBed.createComponent(HomePage);
17      component = fixture.componentInstance;
18      fixture.detectChanges();
19    }));
20
21    it('should create', () => {
22      expect(component).toBeTruthy();
23    });
24  });
```

PGY4121 Estructura General

- Caso de prueba individual

```
5
6   describe('HomePage', () => {
7     let component: HomePage;
8     let fixture: ComponentFixture<HomePage>;
9
10    beforeEach(async(() => {
11      TestBed.configureTestingModule({
12        declarations: [ HomePage ],
13        imports: [IonicModule.forRoot()]
14      }).compileComponents();
15
16      fixture = TestBed.createComponent(HomePage);
17      component = fixture.componentInstance;
18      fixture.detectChanges();
19    }));
20
21    it('should create', () => {
22      expect(component).toBeTruthy();
23    });
24  });
```

PGY4121 Estructura General

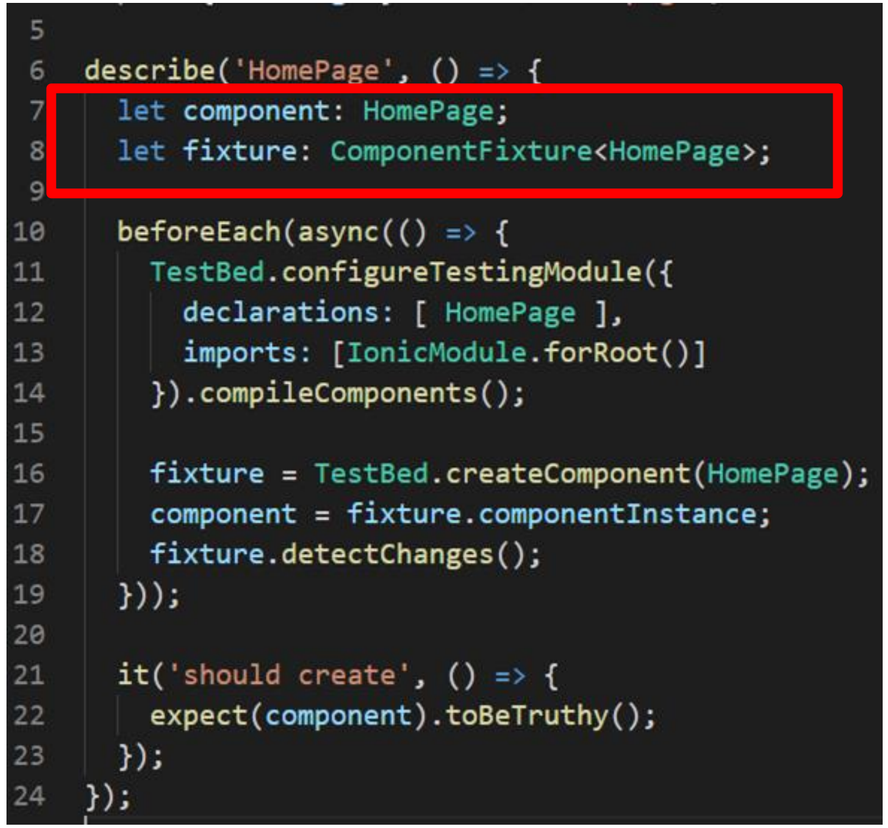
- Tanto él describe y el it tiene un texto descriptivo el cual al ejecutar forma la siguiente frase: ‘HomePage should create’

```
6  describe 'HomePage', () => {
7    let component: HomePage;
8    let fixture: ComponentFixture<HomePage>;
```

```
21   it('should create', () => {
22     expect(component).toBeTruthy();
23   });
24 });
```

PGY4121 Estructura General

- Declaró el componente a testear “let component: HomePage” y declaró el componente bajo prueba mediante el ComponentFixture<HomePage>



```
5
6   describe('HomePage', () => {
7     let component: HomePage;
8     let fixture: ComponentFixture<HomePage>;
9
10    beforeEach(async(() => {
11      TestBed.configureTestingModule({
12        declarations: [ HomePage ],
13        imports: [IonicModule.forRoot()]
14      }).compileComponents();
15
16      fixture = TestBed.createComponent(HomePage);
17      component = fixture.componentInstance;
18      fixture.detectChanges();
19    }));
20
21    it('should create', () => {
22      expect(component).toBeTruthy();
23    });
24  });
```

PGY4121 Funcionamiento TestBed

- Se utiliza el bloque de configuración beforeEach para Configurar el módulo a probar mediante configureTestingModule

```
beforeEach(async(() => {
  TestBed.configureTestingModule({
    declarations: [ HomePage ],
    imports: [IonicModule.forRoot()]
  }).compileComponents();
```

=

```
@NgModule({
  imports: [
    CommonModule,
    FormsModule,
    IonicModule,
    HomePageRoutingModule
  ],
  declarations: [HomePage]
})
export class HomePageModule {}
```

- Finalmente que compile las definiciones con compileComponents()

PGY4121 Funcionamiento TestBed

- Se inyectan componentes mediante **createComponent()** y se obtiene su instancia mediante la propiedad **componentInstance** y se detectan los cambios del componente mediante **detectChanges**

```
beforeEach(async(() => {
  TestBed.configureTestingModule({
    declarations: [ HomePage ],
    imports: [IonicModule.forRoot()]
  }).compileComponents();

  fixture = TestBed.createComponent(HomePage);
  component = fixture.componentInstance;
  fixture.detectChanges();
}));
```

Testeando Servicios

DuocUC®

PGY4121 Servicios Básicos

- Contexto
  - Se presenta un servicio con un método que toma una variedad de impuestos y que calcula el pago neto de los impuestos service = new PayrollService(taxServiceSpy);
  - Se tiene un spy (mock object) llamado TaxService creado de forma manual mediante Jasmin por su metodo createSpyObj
- Objetivo del servicio:
  - Generar una nómina de sueldo con los descuentos respectivos (EEUU)

PGY4121 Servicios Básicos

» Resultado código Test

Se crea el objeto simulado en el código de configuración del test

Se instancia el servicio de cálculo de la nómina de sueldo manualmente

Prueba a ejecutar

```
import { PayrollService } from './payroll.service';

describe('PayrollService', () => {
  let service: PayrollService;
  let taxServiceSpy;

   beforeEach(() => {
     taxServiceSpy = jasmine.createSpyObj('TaxService', {
       federalIncomeTax: 0,
       stateIncomeTax: 0,
       socialSecurity: 0,
       medicare: 0
     });
     service = new PayrollService(taxServiceSpy);
   });

   describe('net pay calculations', () => {
     ...
   });
});
```

PGY4121 Servicios Básicos

» Avanzando un poco más

Se crea el objeto simulado en el código de configuración del test

Configurar el módulo a probar

Prueba a ejecutar mediante inyección

Prueba a ejecutar mediante instancia manual

```
import { TestBed, inject } from '@angular/core/testing';

import { PayrollService } from './payroll.service';
import { TaxService } from './tax.service';

describe('PayrolService', () => {
  let taxServiceSpy;

  beforeEach(() => {
    taxServiceSpy = jasmine.createSpyObj('TaxService', {
      federalIncomeTax: 0,
      stateIncomeTax: 0,
      socialSecurity: 0,
      medicare: 0
    });
    TestBed.configureTestingModule({
      providers: [
        PayrollService,
        { provide: TaxService, useValue: taxServiceSpy }
      ]
    });
  });

  it('does some test where it is injected',
    inject([PayrollService], (service: PayrollService) => {
      expect(service).toBeTruthy();
    })
  );

  it('does some test where it is manually built', () => {
    const service = new PayrollService(taxServiceSpy);
    expect(service).toBeTruthy();
  });
});
```

DuocUC®

PGY4121 Servicios HTTP

» Servicio de Track mediante solicitud HTTP

1.- Se importan componentes y módulos

```
import {
  HttpBackend,
  HttpClient
} from '@angular/common/http';
import {
  HttpTestingController,
  HttpClientTestingModule
} from '@angular/common/http/testing';
import { TestBed, inject } from '@angular/core/testing';

import { IssTrackingDataService } from './iss-tracking-data.service';
```

PGY4121 Servicios HTTP

» Servicio de Track mediante solicitud HTTP

2.- Se crean los componentes a usar y se inicia el código de configuración mediante el beforeEach

```
describe('IssTrackingDataService', () => {
  let httpClient: HttpClient;
  let httpTestingController: HttpTestingController;
  let issTrackingDataService: IssTrackingDataService;

  beforeEach(() => {
    TestBed.configureTestingModule({
      imports: [HttpClientTestingModule],
      providers: [
        IssTrackingDataService
      ]
    });

    httpClient = TestBed.get(HttpClient);
    httpTestingController = TestBed.get(HttpTestingController);
    issTrackingDataService = new IssTrackingDataService(httpClient);
  });
```

PGY4121 Servicios HTTP

» Servicio de Track mediante solicitud HTTP

3.- En el bloque de configuración se configura el módulo y los objetos simulados (mock object) mediante TestBed

```
beforeEach(() => {
  TestBed.configureTestingModule({
    imports: [HttpClientTestingModule],
    providers: [
      IssTrackingDataService
    ]
  });

  httpClient = TestBed.get(HttpClient);
  httpTestingController = TestBed.get(HttpTestingController);
  issTrackingDataService = new IssTrackingDataService(httpClient);
});
```

PGY4121 Servicios HTTP

» Servicio de Track mediante solicitud HTTP

4.- Se ejecuta una prueba si se crea correctamente el servicio inyectando los objetos, y mediante la instrucción **expect** se espera que sea true el objeto service

```
it('exists', inject([IssTrackingDataService], (service: IssTrackingDataService) =
    expect(service).toBeTruthy();
}));
```

PGY4121 Servicios HTTP

» El código completo se puede observar en: [Testing HTTP Data Services](#) de la documentación oficial de Ionic

```
import {
  HttpBackend,
  HttpClient
} from '@angular/common/http';
import {
  HttpTestingController,
  HttpClientTestingModule
} from '@angular/common/http/testing';
import { TestBed, inject } from '@angular/core/testing';

import { IssTrackingDataService } from './iss-tracking-data.service';

describe('IssTrackingDataService', () => {
  let httpClient: HttpClient;
  let httpTestingController: HttpTestingController;
  let issTrackingDataService: IssTrackingDataService;

  beforeEach(() => {
    TestBed.configureTestingModule({
      imports: [HttpClientTestingModule],
      providers: [
        IssTrackingDataService
      ]
    });

    httpClient = TestBed.get(HttpClient);
    httpTestingController = TestBed.get(HttpTestingController);
    issTrackingDataService = new IssTrackingDataService(httpClient);
  });

  it('exists', inject([IssTrackingDataService], (service: IssTrackingDataService) =
      expect(service).toBeTruthy();
  }));
```

```
  describe('location', () => {
    it('gets the location of the ISS now', () => {
      issTrackingDataService.location().subscribe(x => {
        expect(x).toEqual({ longitude: -138.1719, latitude: 44.4423 });
      });
      const req = httpTestingController.expectOne(
        'http://api.open-notify.org/iss-now.json'
      );
      expect(req.request.method).toEqual('GET');
      req.flush({
        iss_position: { longitude: '-138.1719', latitude: '44.4423' },
        timestamp: 1525950644,
        message: 'success'
      });
      httpTestingController.verify();
    });
  });
});
```

DuocUC®

Profundiza más!!

Consultar la documentación oficial:

https://ionicframework.com/docs/angular/testing

Testing Scrips:

https://ionicframework.com/docs/angular/testing#testing-scripts

Inyección de componentes Angular: https://medium.com/angular-chile/inyección-de-componentes-y-directivas-en-angular-6ae75f64be66

Que es inyección:

https://es.wikipedia.org/wiki/Inyección_de_dependencias

DuocUC®