# Testing

## *Pruebas en Angular Material*

CertiDevs

# Índice de contenidos

# 1. Testing con Angular Material

Los **Angular Material Harnesses** son clases de utilidad que facilitan la interacción con componentes de Angular Material en las pruebas.

Sitio oficial de Angular Material Harnesses

En este ejemplo, vamos a crear pruebas para un componente que utiliza una tabla de Angular Material (`MatTable`).

Supongamos que tenemos un **componente** llamado `UsersTableComponent` que muestra una tabla de usuarios utilizando el componente `MatTable` de Angular Material.

Archivo `users-table.component.ts`:

```
import { Component } from '@angular/core';

export interface User {
    id: number;
    name: string;
    email: string;
}

@Component({
    selector: 'app-users-table',
    template: `
        <table mat-table [dataSource]="users" class="mat-elevation-z8">
            <ng-container matColumnDef="id">
                <th mat-header-cell *matHeaderCellDef>ID</th>
                <td mat-cell *matCellDef="let user">{{ user.id }}</td>
            </ng-container>

            <ng-container matColumnDef="name">
                <th mat-header-cell *matHeaderCellDef>Name</th>
                <td mat-cell *matCellDef="let user">{{ user.name }}</td>
            </ng-container>

            <ng-container matColumnDef="email">
                <th mat-header-cell *matHeaderCellDef>Email</th>
                <td mat-cell *matCellDef="let user">{{ user.email }}</td>
            </ng-container>

            <tr mat-header-row *matHeaderRowDef="displayedColumns"></tr>
            <tr mat-row *matRowDef="let row; columns: displayedColumns"></tr>
        </table>
    `,
})
export class UsersTableComponent {
    displayedColumns: string[] = ['id', 'name', 'email'];
    users: User[] = [
```

```
        { id: 1, name: 'John Doe', email: 'john.doe@example.com' },
        { id: 2, name: 'Jane Doe', email: 'jane.doe@example.com' },
    ];
}
```

Para probar este componente utilizando Angular Material Harnesses, siga estos pasos:

## 1.1. dependencias necesarias

Tener instalado angular material:

```
npm install @angular/material @angular/cdk
```

Importar las dependencias necesarias en el archivo de prueba `users-table.component.spec.ts`:

```
import { ComponentFixture, TestBed } from '@angular/core/testing';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';
import { MatTableModule } from '@angular/material/table';
import { MatTableHarness } from '@angular/material/table/testing';
import { UsersTableComponent } from './users-table.component';
```

Configurar el entorno de prueba utilizando `TestBed` y `ComponentFixture`:

```
let component: UsersTableComponent;
let fixture: ComponentFixture<UsersTableComponent>;

beforeEach(async () => {
    await TestBed.configureTestingModule({
        imports: [MatTableModule, BrowserAnimationsModule],
        declarations: [UsersTableComponent],
    }).compileComponents();

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

Registre `MatTableHarness` como una clase de harness que se utilizará en las pruebas:

```
import { TestbedHarnessEnvironment } from '@angular/cdk/testing/testbed';
import { HarnessLoader } from '@angular/cdk/testing';

let loader: HarnessLoader;

beforeEach(() => {
    loader = TestbedHarnessEnvironment.loader(fixture);
```

```
});
```

Escribir pruebas utilizando `MatTableHarness` para interactuar con el componente `MatTable`.

En este caso, se probará que la tabla tenga la cantidad correcta de filas y columnas, y que los datos de las celdas sean correctos:

```
it('should have the correct number of rows and columns', async () => {
    const tableHarness = await loader.getHarness(MatTableHarness);
    const rows = await tableHarness.getRows();
    const headerRows = await tableHarness.getHeaderRows();
    const footerRows = await tableHarness.getFooterRows();

    expect(rows.length).toBe(2);
    expect(headerRows.length).toBe(1);
    expect(footerRows.length).toBe(0);
});

it('should have the correct cell data', async () => {
    const tableHarness = await loader.getHarness(MatTableHarness);
    const rows = await tableHarness.getRows();
    const firstRowCells = await rows[0].getCells();
    const secondRowCells = await rows[1].getCells();

    expect(firstRowCells.length).toBe(3);
    expect(await firstRowCells[0].getText()).toBe('1');
    expect(await firstRowCells[1].getText()).toBe('John Doe');
    expect(await firstRowCells[2].getText()).toBe('john.doe@example.com');

    expect(secondRowCells.length).toBe(3);
    expect(await secondRowCells[0].getText()).toBe('2');
    expect(await secondRowCells[1].getText()).toBe('Jane Doe');
    expect(await secondRowCells[2].getText()).toBe('jane.doe@example.com');
});
```

En este ejemplo, utilizamos `MatTableHarness` para interactuar con el componente `MatTable`.

Las pruebas verifican que la tabla tenga la cantidad correcta de filas y columnas y que los datos de las celdas sean los esperados.

Al utilizar Angular Material Harnesses, se simplifica la interacción con los componentes de Angular Material en las pruebas y se proporciona una API consistente y fácil de usar.

Puede seguir un enfoque similar para probar otros componentes de Angular Material utilizando sus respectivos Harnesses.

Ver más ejemplos de Angular Material Harnesses