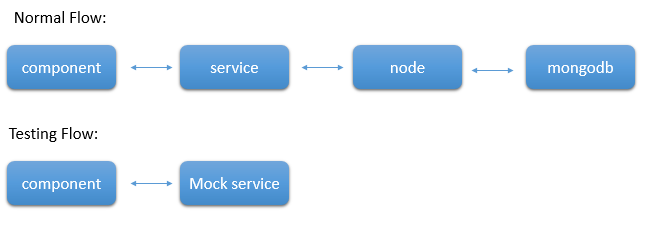
If the component is using a service, then when you are testing the component, you need the service as well. If the service is connecting to a node backend, which is in turn connecting to a mongodb, then we need the entire setup just to test a component.

Mocking is the practice of replacing the actual service object with a dummy or mock service object. The mock service will be returning some hard coded dummy data instead of the actual data from the server.



Thus you don't need the entire set up to test the component alone.

To mock a service, we need to create a dummy service. Let's say that our original service (MessageSevice) has a getName() and it returns an observable of a string, which is returned from the back-end server. Then your mock service will look like:

**mock.service.ts:**

1. getName():Observable<string> {
2. return of("Hello World");
3. }

As you saw in Observables, ***of*** is used to convert a value into an Observable. In the component, where the service is being used, you need to update its spec file. For example, if the message component is using the service, then you need to update the providers of the **message.component.spec.ts** file as shown below:

1. providers:[{provide:MessageService,useClass:MockService}]

Thus during testing, MockSevice object will be injected instead of the MessageService object. Thus you can mock a service and test the component alone.

If your application uses routes, they should be included in the app.component.spec.ts. Else, the test cases will fail.

The below lines need to be included in the app.component.spec.ts:

1. import {APP\_BASE\_HREF} from '@angular/common';
3. describe('AppComponent', () => {
4. const routes:Routes=[
5. {path:'message',component:MessageComponent}
6. ]
7. beforeEach(async(() => {
8. TestBed.configureTestingModule({
9. declarations: [
10. AppComponent,
11. MessageComponent,
12. HelloPipe
13. ],
14. imports:[RouterModule.forRoot(routes)],
15. providers: [{provide: APP\_BASE\_HREF, useValue : '/' }]
16. }).compileComponents();