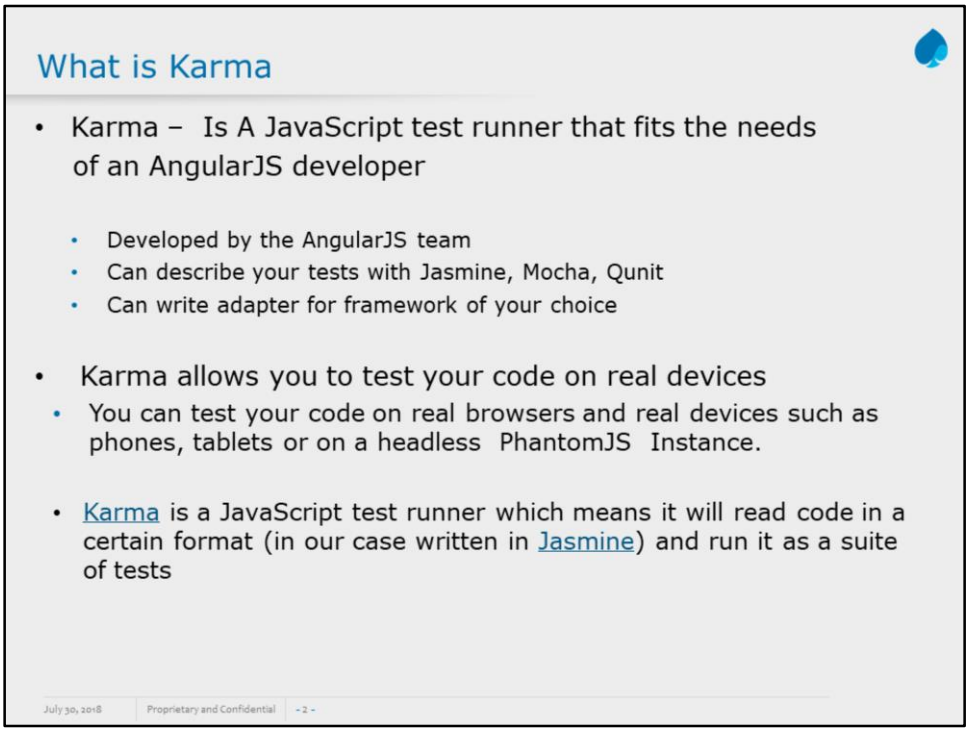
Instructor Notes:

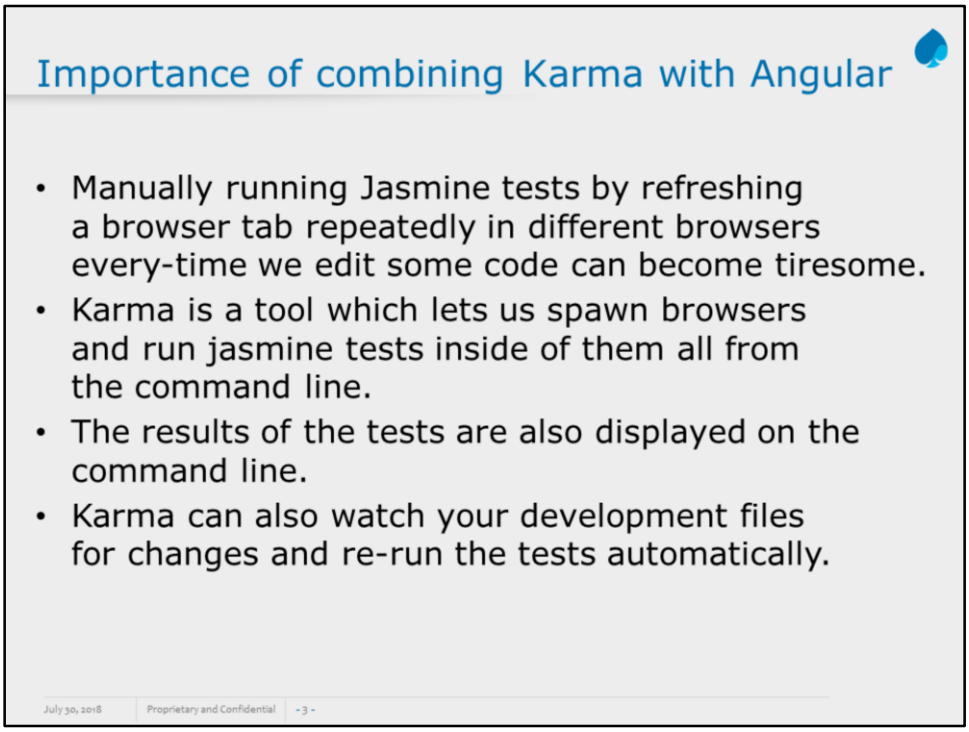
Add instructor notes here.

Instructor Notes:



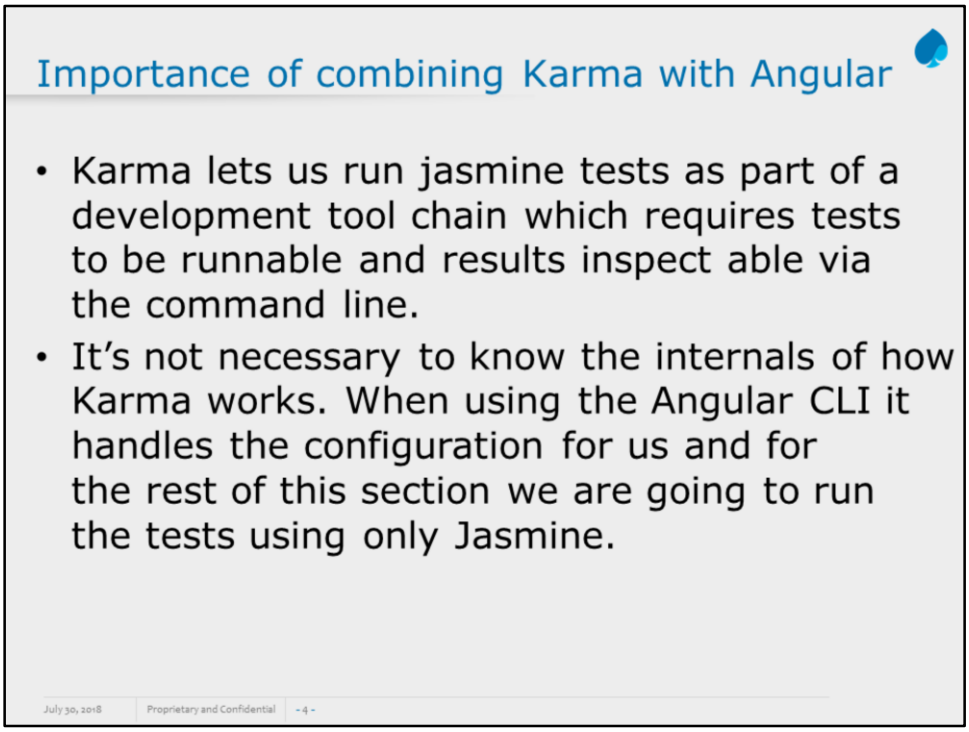
- Unit Test
  - In unit tests component is tested in isolation without external recourses like file system and database and API
  - In angular terms testing angular component without template ,recourses, browser , API.
  - Ex angular component is using service to talk to API end point .then instead of service we can use Fake service instance.
  - Will focus functionality of component. Or Fake router.
- Integration Test
  -We test the component with external resources like file system ,database, and API end point
- End To End Test
  –we test application in the browser -very slow and fragile.

Instructor Notes:

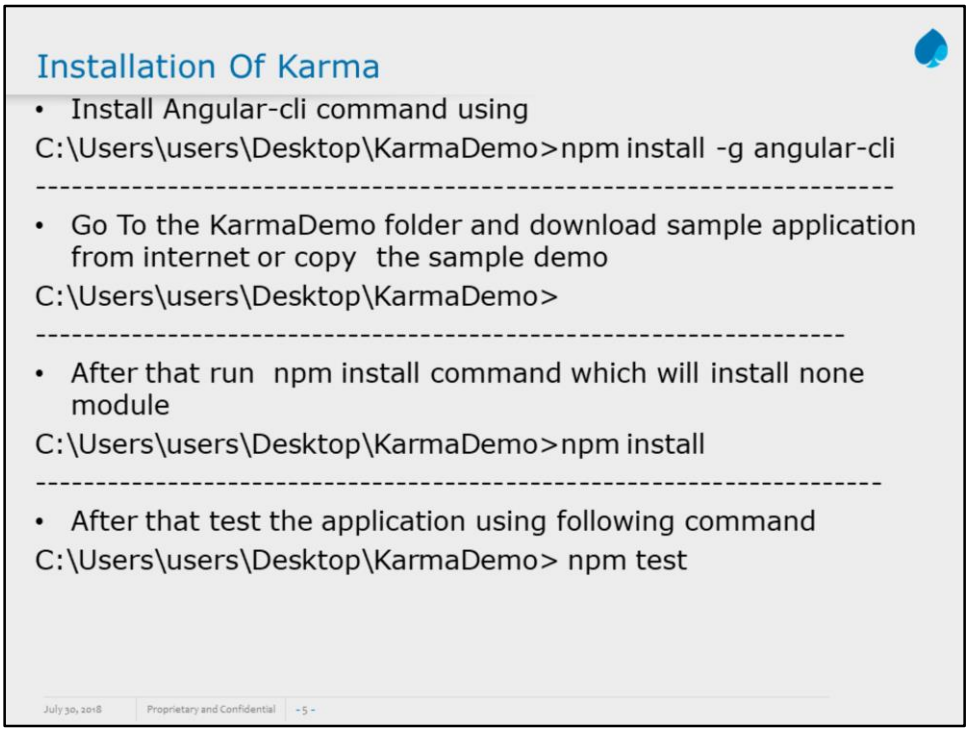


Note:- Naming convention to be followed for writing angular unit test cases.
-If component name is **"welcome.component.ts"** then test file name should be "welcome.component.spec.js
- We use angular-cli ng-test command to run the unit test cases.

Instructor Notes:



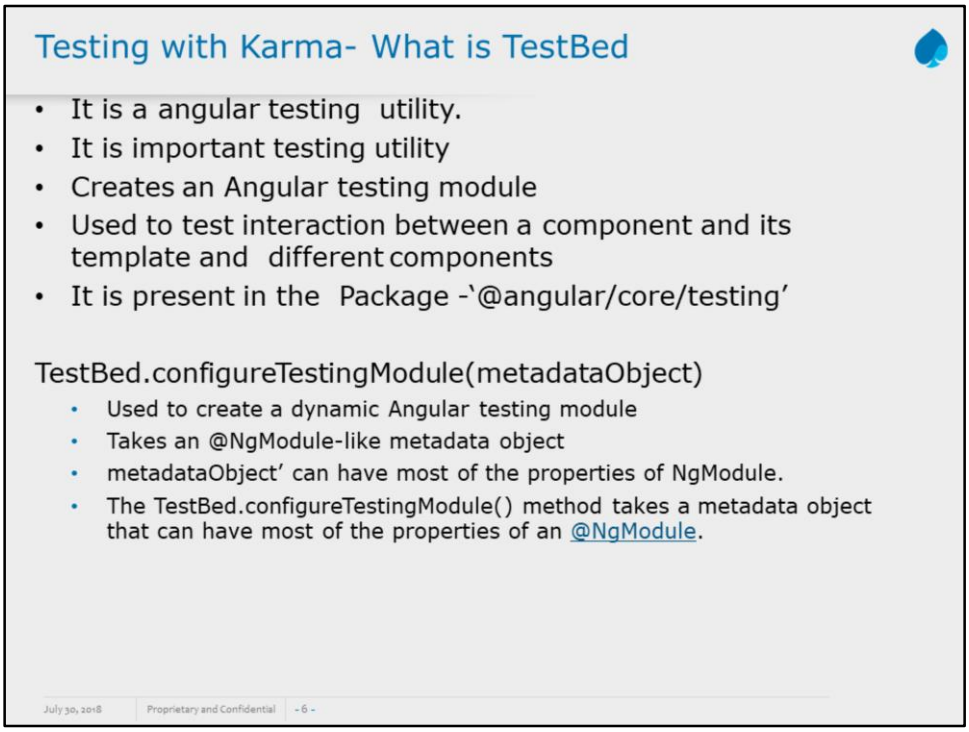
Instructor Notes:



.Set up Karma
This setup is done through the common interface of the Karma configuration file called *karma.conf.js*.
This file will tell Karma which locations the tests are in, which reporting mechanism to use, and which browser to use to execute the tests. Here is an example Karma config file from the Angular CLI project:

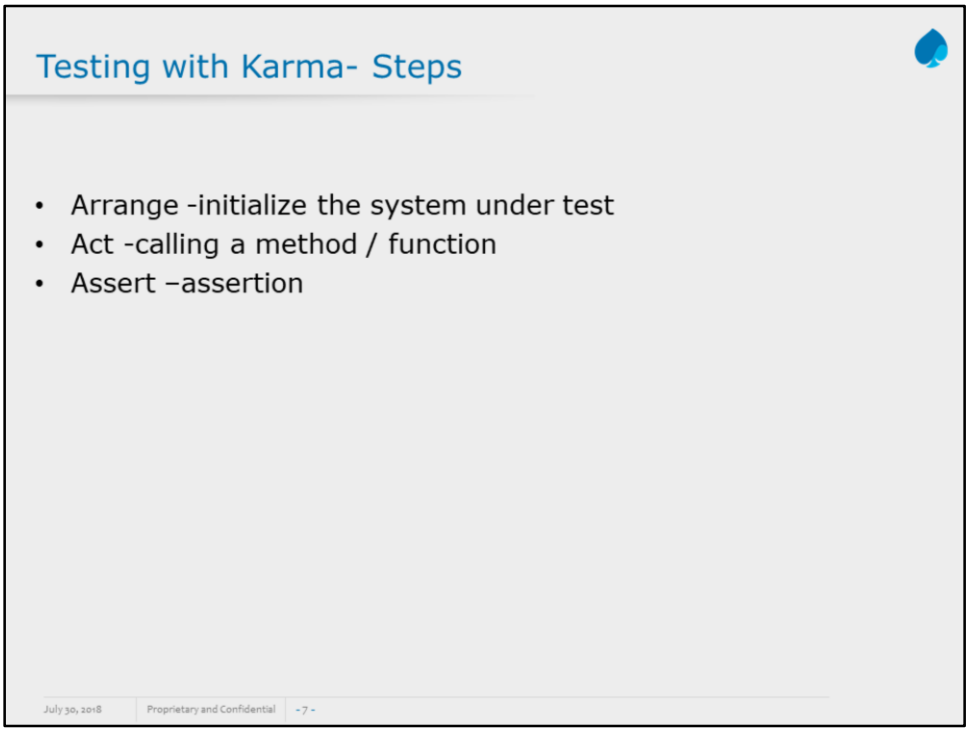
```
///...
basePath: '',
frameworks: ['jasmine', 'angular-cli'],
plugins: [
  require('karma-jasmine'),
  require('karma-chrome-launcher'),
  require('karma-remap-istanbul'),
  require('angular-cli/plugins/karma')
],
files: [
  { pattern: './src/test.ts', watched: false }
],
preprocessors: {
  './src/test.ts': ['angular-cli']
},
remapIstanbulReporter: {
  reports: {
    html: 'coverage',
    lcovonly: './coverage/coverage.lcov'
  }
},
angularCli: {
  config: './angular-cli.json',
  environment: 'dev'
},
///...
```

Instructor Notes:

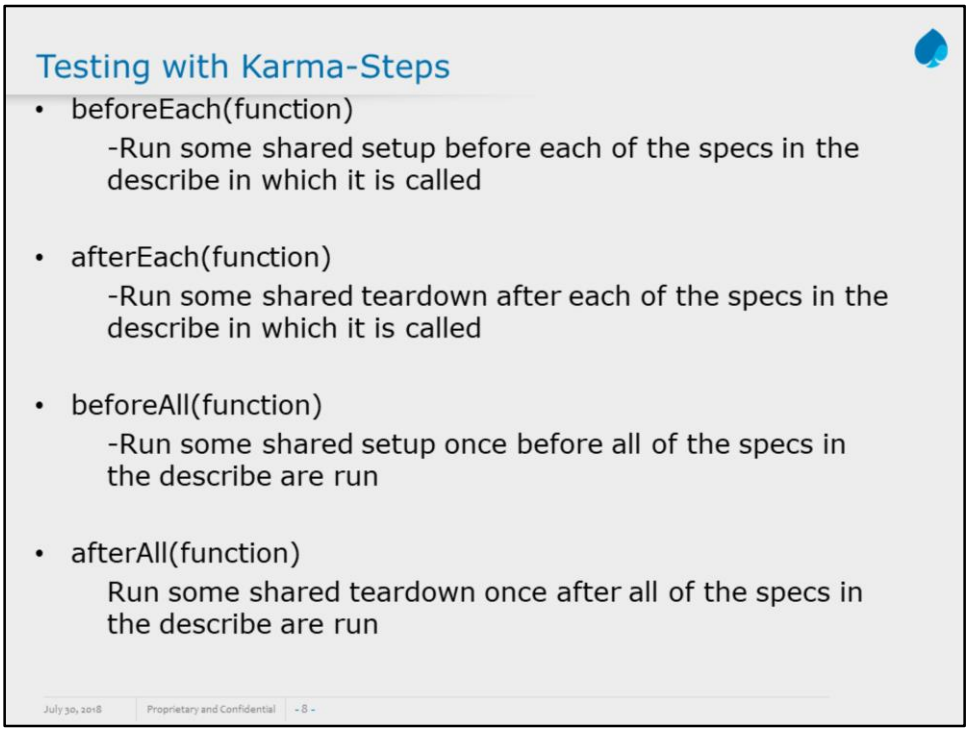


In Angular 2 in order to test any component we need to load it into a module beforehand.
This is done (in part) by a handy built-in helper class called *TestBed*.

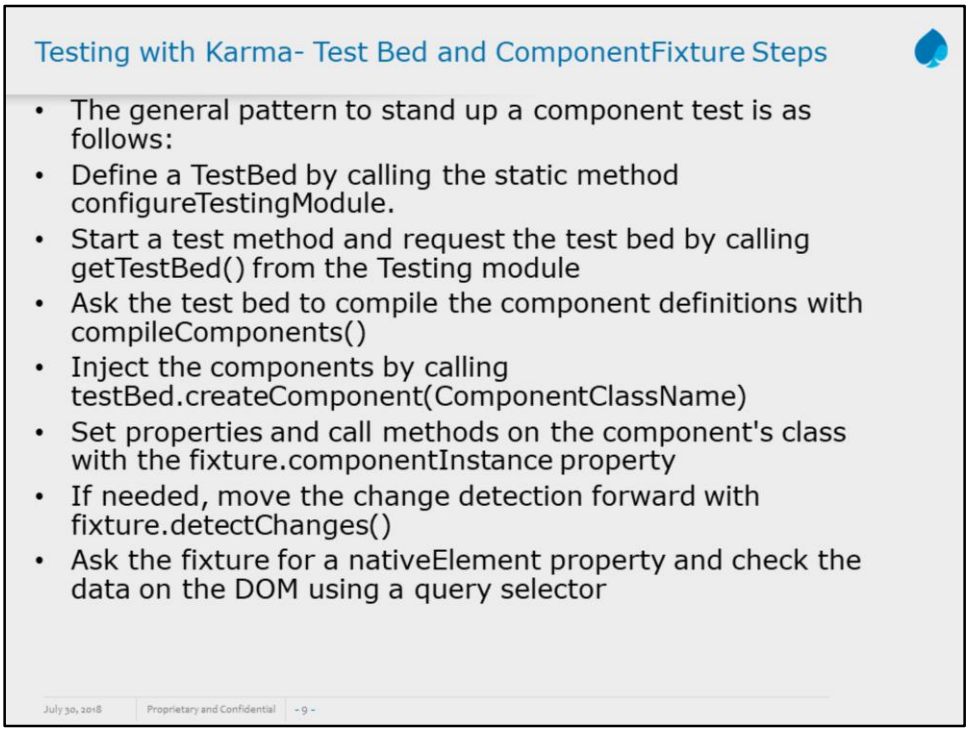
Instructor Notes:



Instructor Notes:

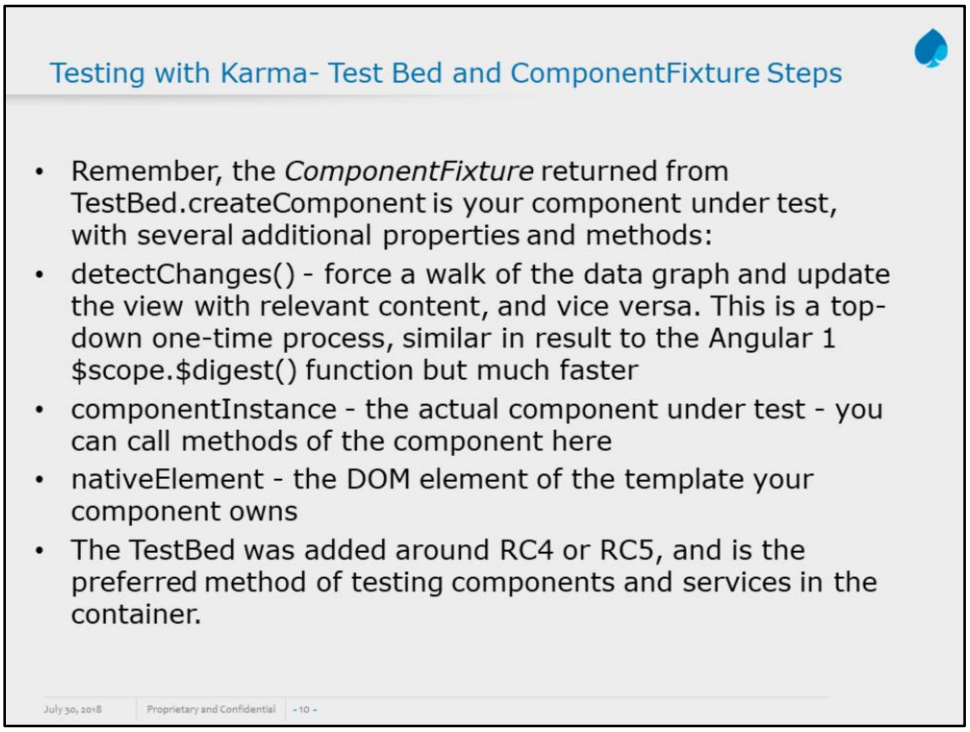


Instructor Notes:

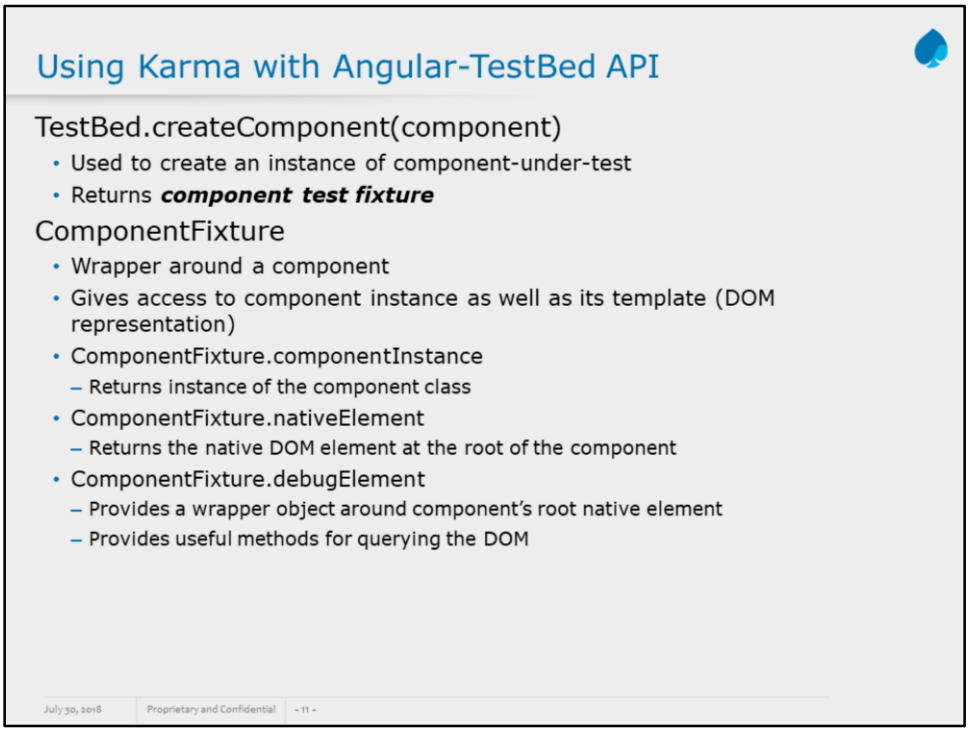


In Angular 2 in order to test any component we need to load it into a module beforehand.
This is done (in part) by a handy built-in helper class called *TestBed*.

Instructor Notes:



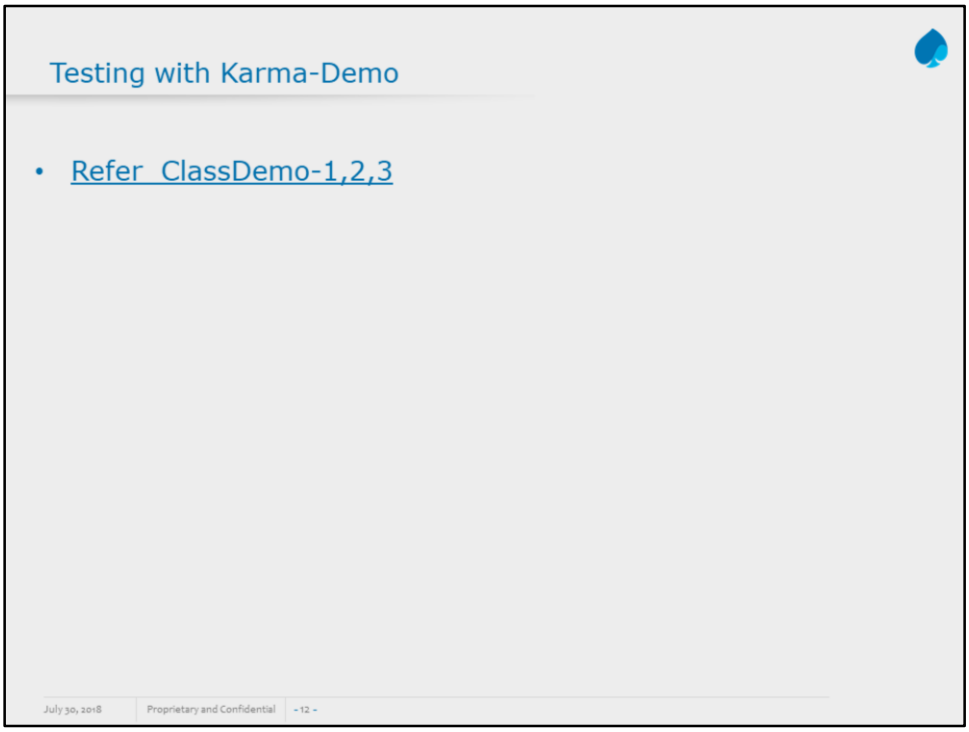
Instructor Notes:



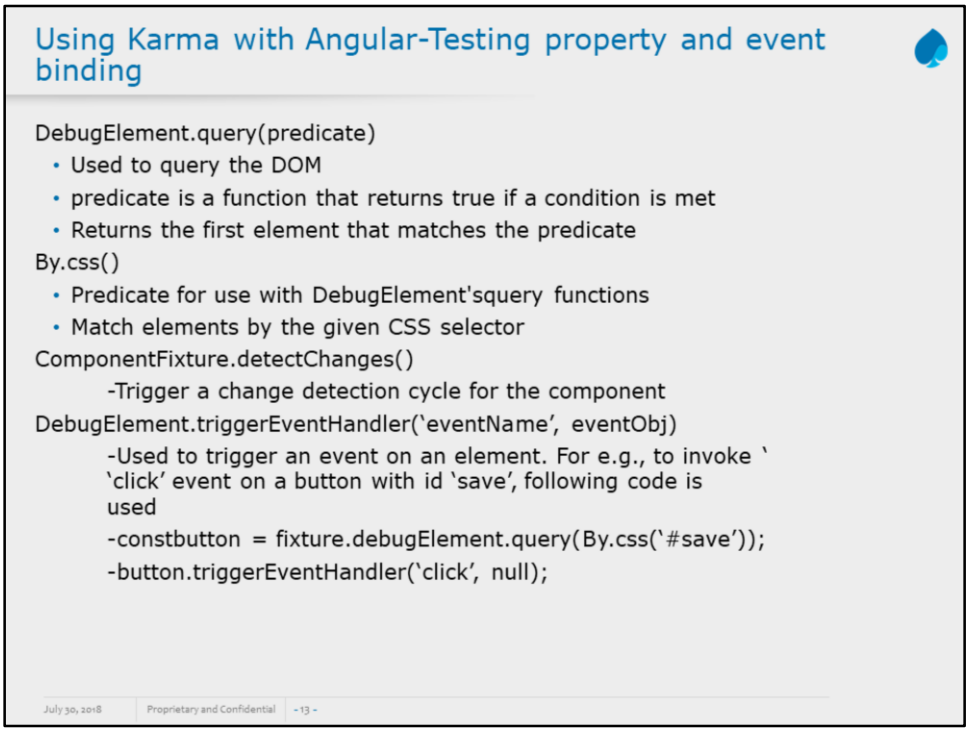

```
import { async, TestBed } from '@angular/core/testing';
import { SomeComponent } from './some.component';
beforeEach(() => {
 TestBed.configureTestingModule({ declarations: [ SomeComponent ],
imports: [ // HttpModule, etc. ], providers: [ // { provide: ServiceA, useClass:
TestServiceA } ] }); });

it('should do something', async(() =>
{
 // Overrides here, if you need them
TestBed.overrideComponent(SomeComponent, { set: { template: '<div>Overridden
template here</div>' // ... } });

TestBed.compileComponents().then(() => { const fixture =
TestBed.createComponent(SomeComponent);

// Access the dependency injected component instance
const app = fixture.componentInstance; expect(app.something).toBe('something');

 // Access the element
const element = fixture.nativeElement;

 // Detect changes as necessary
fixture.detectChanges();
 expect(element.textContent).toContain('something'); }); }));
```

A fixture is a wrapper for a component *and* it's template.
We create an instance of a component fixture through the TestBed, this injects the AuthService into the component constructor.
We can find the actual *component* from the componentInstance on the fixture.
We can get resolve dependencies using the TestBed injector by using the get function.

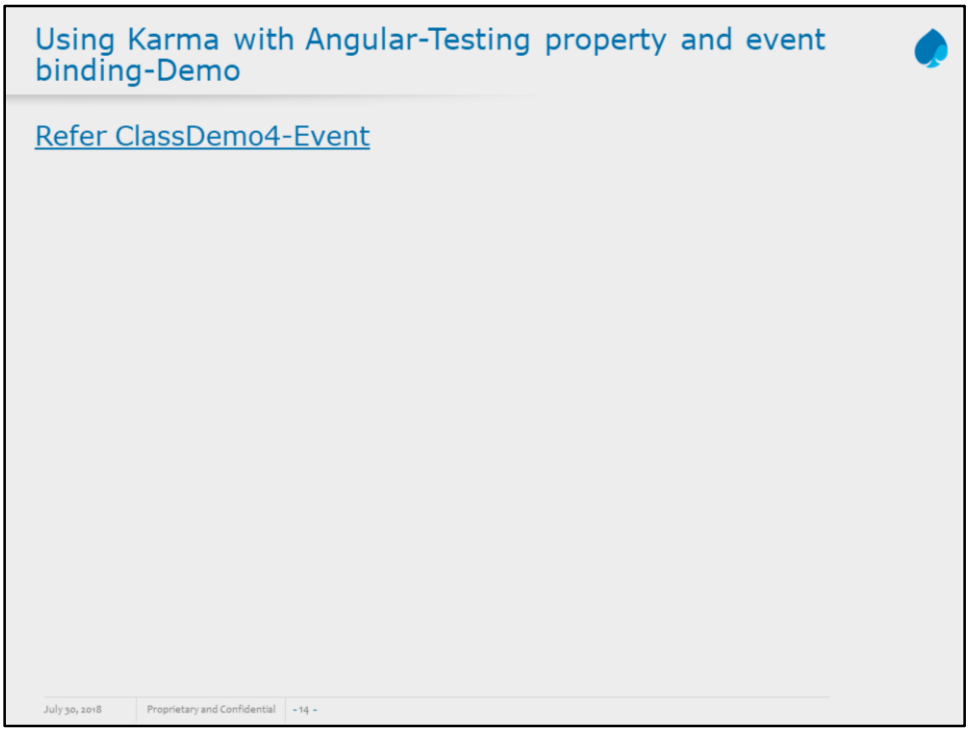
Instructor Notes:



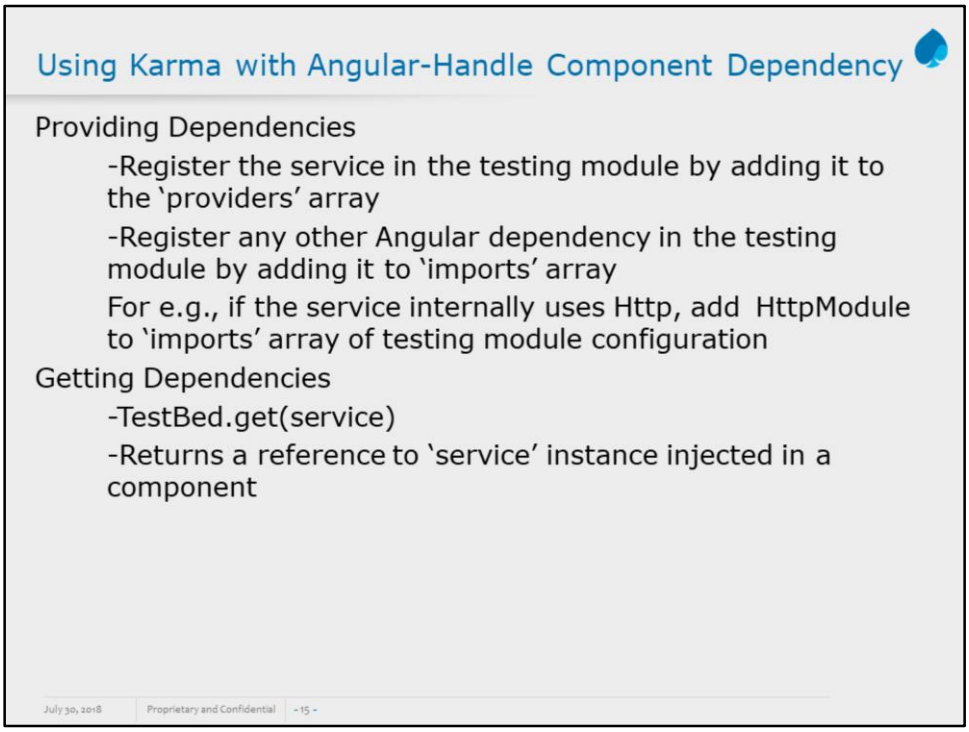
Instructor Notes:



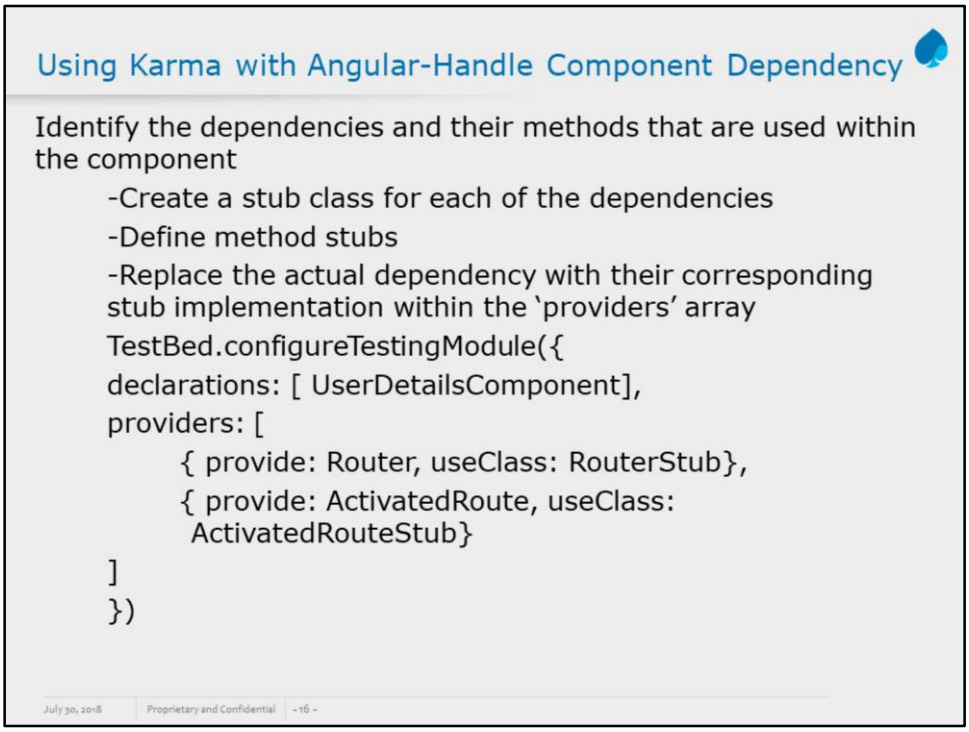
Instructor Notes:



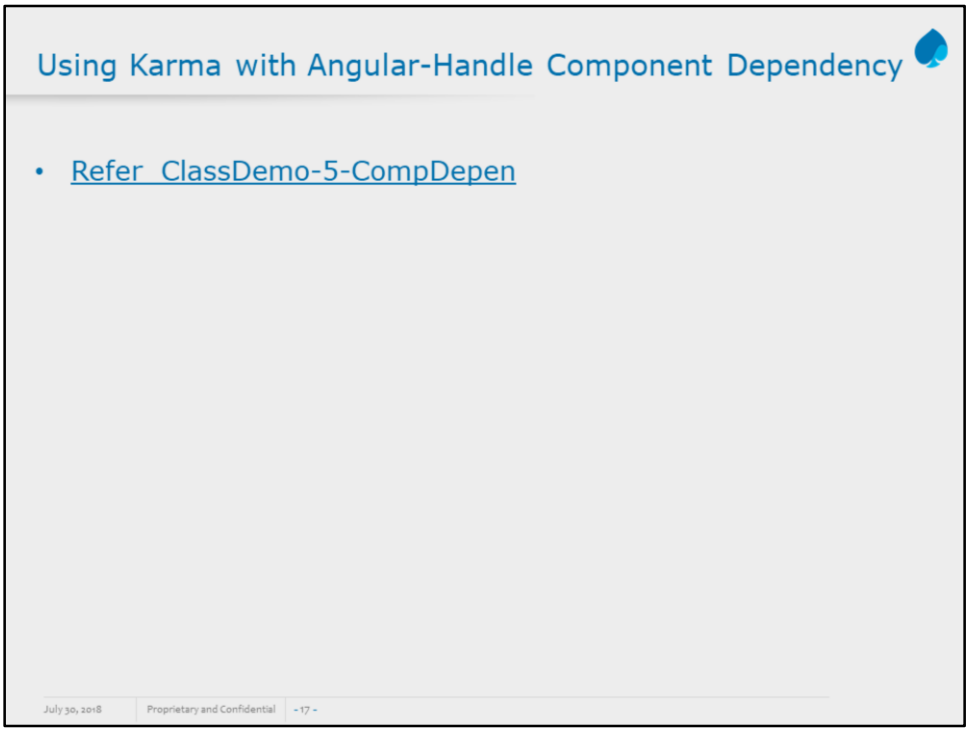
Instructor Notes:



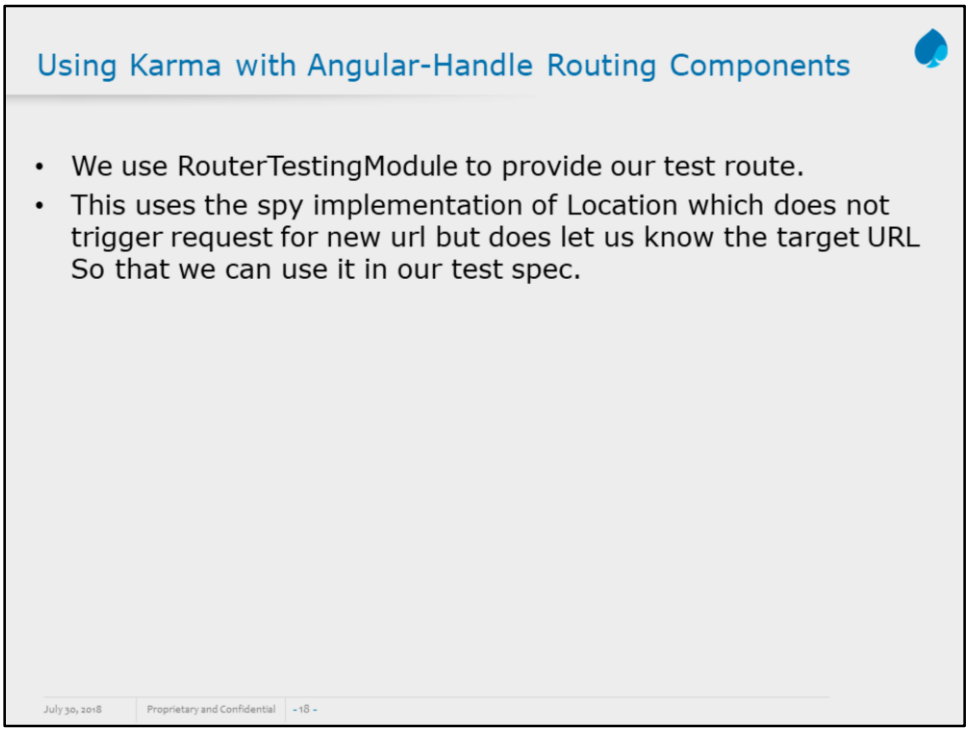
Instructor Notes:



Instructor Notes:



Instructor Notes:



Instructor Notes: