

ANGULAR TESTING IN VIGO NG

INHALT

- Tools
- Angular Testing
- Types of Tests
- VIGO NG
Testing

TOOLS

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]:
2 09 04 2019 21:03:33.732:INFO [launcher]:
3 09 04 2019 21:03:33.741:INFO [launcher]:
4 09 04 2019 21:03:59.049:WARN [karma]:
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]:
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]:
3 09 04 2019 21:03:33.741:INFO [launcher]:
4 09 04 2019 21:03:59.049:WARN [karma]:
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]:
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]: Launching browser ChromeCustom with unlimited concurrency
3 09 04 2019 21:03:33.741:INFO [launcher]:
4 09 04 2019 21:03:59.049:WARN [karma]:
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]:
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]: Launching browser ChromeCustom with unlimited concurrency
3 09 04 2019 21:03:33.741:INFO [launcher]: Starting browser Chrome
4 09 04 2019 21:03:59.049:WARN [karma]:
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]:
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]: Launching browser ChromeCustom with unlimited concurrency
3 09 04 2019 21:03:33.741:INFO [launcher]: Starting browser Chrome
4 09 04 2019 21:03:59.049:WARN [karma]: No captured browser, open http://localhost:9876/
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]:
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]: Launching browser ChromeCustom with unlimited concurrency
3 09 04 2019 21:03:33.741:INFO [launcher]: Starting browser Chrome
4 09 04 2019 21:03:59.049:WARN [karma]: No captured browser, open http://localhost:9876/
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]: Connected on socket ... with id 0
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

KARMA

```
1 09 04 2019 21:03:33.731:INFO [karma]: Karma v1.7.1 server started at http://0.0.0.0:9876/
2 09 04 2019 21:03:33.732:INFO [launcher]: Launching browser ChromeCustom with unlimited concurrency
3 09 04 2019 21:03:33.741:INFO [launcher]: Starting browser Chrome
4 09 04 2019 21:03:59.049:WARN [karma]: No captured browser, open http://localhost:9876/
5 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]: Connected on socket ... with id 0
6 09 04 2019 21:03:59.500:INFO [HeadlessChrome 73.0.3683 (Linux 0.0.0)]: Executed 81 of 424 SUCCESS (0 secs / 18.725 secs)
```

Speaker notes

- 1 Zeile: Karma server
- 2 Zeile: Karma startet einen browser als client
- 3 Zeile: Browser startet
- 4 Zeile: Browser capturing, debugging in chrome etc.
- 5 Zeile: Socket connection für server -> browser crosstalk
- 6 Zeile: Tests running
- Test Runner
- Debugging
- Browser und Plattform unabhängig

JASMINE

```
1 describe("a suite", () => {
2   it("should add 2 and 2 to 4", () => {
3     expect(2 + 2).toBe(4);
4   });
5
6   it("should concatenate strings", () => {
7     const a = 'someName';
8     expect(`this is my name: ${a}`).toBe('this is my name: someName');
9   });
10 });
```

Speaker notes

- Fast
- BDD for JavaScript
- Spying

JASMINE MATCHERS

```
1 // not.(other-matcher)
2 expect(something).not.toBe(true);
3
4 // equals
5 expect(something).toBe(true);
6
7 // equals float
8 expect(something).toBeCloseTo(expected, precision);
9 expect(number).toBeCloseTo(42.2, 3);
10
11 //contains
12 expect(array).toContain(anElement);
13
14 //error
15 expect(aFunction()).toThrow(anError);
```

TS-MOCKITO

```
1 // from ts-mockito docs:
2 // Creating mock
3 let mockedFoo:Foo = mock(Foo);
4
5 // stub method before execution
6 when(mockedFoo.getBar(3)).thenReturn('three');
7
8 // Getting instance
9 let foo:Foo = instance(mockedFoo);
10
11 // prints three
```

Speaker notes

- When to stub methods doesn't matter
- pass instance to logic
- use mock to stub and verify
- can spy on real objects
- for java developers it's basically the same, except for the instance

ANGULAR TESTING

TESTBED & TESTING MODULE

```
1 let component: BannerComponent;
2 let fixture: ComponentFixture<BannerComponent>;
3 let h1: HTMLElement;
4
5 beforeEach(() => {
6   TestBed.configureTestingModule({
7     declarations: [BannerComponent],
8     providers: [],
9     imports: []
10  });
11
12  fixture = TestBed.createComponent(BannerComponent);
13  component = fixture.componentInstance;
14  h1 = fixture.nativeElement.querySelector('h1');
```

Speaker notes

- Angular Modules for Testing
- Bootstraps Angular

HTTP CLIENT TESTING MODULE

```
1 TestBed.configureTestingModule({  
2     //...  
3     imports: [HttpClientTestingModule]  
4 });
```

```
1 const req = controller.expectOne(`myBasePath/gfs/${beilage.gfId}/beilagen`);  
2  
3 expect(req.request.method).toBe('POST');  
4 const body: FormData = req.request.body;  
5 expect(body.get('art')).toBe(beilage.art);  
6 //...  
7 req.flush('A', {headers: new HttpHeaders()});
```

Speaker notes

- Can be used to make sure requests are correctly sent to be.

TESTBED PROS/CONS

Positives

- Dependency injection
- Importing modules
- Mocking with injection
- Less code in tests
- Tests module configuration

Negatives

- Performance

TYPES OF TESTS

SHALLOW TESTS

- Shallow-render
- Template testing
- Component bootstrap testing
- NgOnInit template set-up testing

Speaker notes

- Shallow-render mit schema 'NO_ERRORS_SCHEMA'
- soll nicht die Controller Logik testen, sondern nur die template

ISOLATED TESTS

- Controller testing
- Isolated from other units
- Mock every dependency (except ZEntity related things)
- No Angular bootstrap

Speaker notes

- real unit-test
- no testbed, no module

SERVICE TESTS

- Isolated from other units
- Mock every dependency (except ZEntity related things)
- TestBed vs non-TestBed tests

Speaker notes

- semi-real unit-test for services
- no testbed, no module

INTEGRATION/E2E TESTS

- Protractor
- Page-object pattern
- Multiple components and services

Speaker notes

- important for catching configuration issues, and wrong use of components, because shallow tests don't cover that.

VIGO NG TESTING

MOTIVATION

- Cumbersome
TestBed
- Big Tests
- Many dependencies

Speaker notes

- Always was the same
setup
- A lot of code for setup only
- Intimidating
- Inconsistent development

SETUP

```
1 // these replace the TestBed.configureTestingModule(...) in be
2 setupServiceTestBuilder(Service);
3 setupShallowTestBuilder(Component);
4 setupIsolatedTestBuilder(Controller);
5
6 //configurable
7 setupShallowTestBuilder(Component, builder => builder
8     .withImports(SomeModule)
9     .withProviders(MyManager, GeschaeftsfallManager)
10    .withForms());
```

Speaker notes

- setup methods call the builder
- the builder include alot of default mocks and dependencies
- alot less code and less cognitive complexity

TESTCONTEXT

```
1 // these replace the TestBed.configureTestingModule(...) in bef
2 forServiceTestContext(context => ...);
3 forShallowTestContext(context => ...);
4 forIsolatedTestContext(context => ...);
5
6 // zentity can be configured above aswell but this also exists
7 // in here create ZEntities and save them to the "Store" (map i
8 setupZentityMock(mockService => ...)
```

Speaker notes

- contains zentityMock
- contains map with all mocks
- has method to get instance
- has component instance and fixture
- has test injector

TESTING COMPONENTS

```
1 describe('BeilagenHinzufuegenDialogComponent (Shallow Test)', () => {
2   let testContext: ShallowTestContext<BeilagenHinzufuegenDialogComponent>
3   let beilagenService: BeilagenService;
4
5   beforeEach(() => {
6     beilagenService = mock(BeilagenService);
7     when(beilagenService.getBeilagenArtEnum()).thenReturn(of([]));
8   });
9
10  setupShallowTestBuilder(BeilagenHinzufuegenDialogComponent, builder =>
11    builder
12      .withProviders({provide: BeilagenService, useValue: instance(beilagenService)}, GfServiceResponseManager)
13      .withDialog()
14  );
15
16  forShallowTestContext<BeilagenHinzufuegenDialogComponent>(context => (testContext = context));
17
18  setupZentityMock(mockService => {
19    const abc = new BasisGeschaeftsfallUI();
20    abc.bestandsystemId = '123';
21    mockService.setAtStore(abc);
22  });
23
24  it('should create', () => {
25    expect(testContext.component).toBeTruthy();
26  });
27 });
28
```

Speaker notes

- shallow and isolated test in the same file, because of angular guidelines
- has test injector

TESTING SERVICES

```
1 describe('HttpMessagesService', () => {
2   let service: HttpMessagesService;
3   let messagingServiceMock: MessagingService;
4
5   beforeEach(() => {
6     messagingServiceMock = mock(MessagingService);
7   });
8
9   setupServiceTestBuilder(HttpMessagesService, builder =>
10     builder.withProviders(
11       {provide: MessagingService, useValue: instance(messagingServiceMock)},
12     )
13   );
14
15   forTestContext(context => {
16     service = context.getInstance(HttpMessagesService);
17   });
18
19   it('should create', () => {
20     expect(service).toBeTruthy();
21   });
22
23   it('should check if meldung is fachlich', () => {
24     const fachlicheMeldung = (service as any).checkIfMsgIsFachlich({
25       id: 'gg',
26       text: 'bla',
27       schweregrad: 'ERROR',
28       marker: ['FACHLICHE_MELDUNG', 'TECHNISCHE_MELDUNG']
```

Speaker notes

- shallow and isolated test in the same file, because of angular guidelines
- has test injector