

Testing HTTP Requests



Brice Wilson

@brice_wilson www.BriceWilson.net



Default Angular Unit Testing Tools

Jasmine

Karma

Angular testing utilities

Angular CLI

npm scripts



Structure of Jasmine Unit Tests

```
describe('DataService Test Suite', () => {
```

```
});
```



Structure of Jasmine Unit Tests

```
describe('DataService Test Suite', () => {  
  beforeEach(() => {  
    // setup code run before each test  
  });  
});
```

```
});
```



Structure of Jasmine Unit Tests

```
describe('DataService Test Suite', () => {  
  beforeEach(() => {  
    // setup code run before each test  
  });  
  it('should do the thing I expect', () => {  
    // execute some code and test result  
  });  
  
});
```



Structure of Jasmine Unit Tests

```
describe('DataService Test Suite', () => {  
  beforeEach(() => {  
    // setup code run before each test  
  });  
  it('should do the thing I expect', () => {  
    // execute some code and test result  
  });  
  afterEach(() => {  
    // teardown code run after each test  
  });  
});
```



Angular HTTP Testing Utilities

HttpClientTestingModule



Angular HTTP Testing Utilities

HttpClientTestingModule

HttpTestingController



Structure of Angular HTTP Unit Tests

```
beforeEach(() => {
```

```
});
```



Structure of Angular HTTP Unit Tests

[illegible]

Structure of Angular HTTP Unit Tests

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



Structure of Angular HTTP Unit Tests

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



Structure of Angular HTTP Unit Tests

```
beforeEach(() => {  
    TestBed.configureTestingModule({  
        imports: [ HttpClientTestingModule ],  
        providers: [ DataService ]  
    });  
    dataService = TestBed.get(DataService);  
    httpTestingController = TestBed.get(HttpTestingController);  
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {
```

```
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
  
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
    let req = httpTestingController.expectOne('/api/books/2');  
  
    });
```




Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
    let req = httpTestingController.expectOne('/api/books/2');  
    // test request here  
  
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
  dataService.getBookById(2)  
    .subscribe(  
      data => { /* test response here */ }  
    );  
  let req = httpTestingController.expectOne('/api/books/2');  
  // test request here  
  
});
```




Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
    let req = httpTestingController.expectOne('/api/books/2');  
    // test request here  
    req.flush(<Book>{  
  
    });  
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
    let req = httpTestingController.expectOne('/api/books/2');  
    // test request here  
    req.flush(<Book>{  
  
    });  
});
```



Structure of Angular HTTP Unit Tests

```
it('should return the correct book', () => {  
    dataService.getBookById(2)  
        .subscribe(  
            data => { /* test response here */ }  
        );  
    let req = httpTestingController.expectOne('/api/books/2');  
    // test request here  
    req.flush(<Book>{  
        title: 'Winnie-the-Pooh',  
        author: 'A. A. Milne'  
    });  
});
```



Demo



Testing HTTP requests and responses



Demo



Testing HTTP errors

