

Testgetriebene Entwicklung mit JavaScript

tobias.bosch@opitz-consulting.com
(@tigbro)



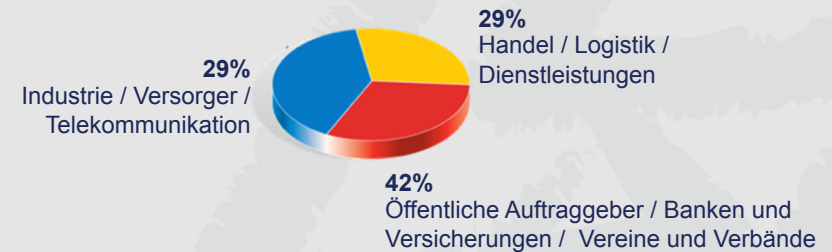
Mission

Wir entwickeln gemeinsam mit allen Branchen Lösungen, die dazu führen, dass sich diese Organisationen besser entwickeln als ihr Wettbewerb.

Unsere Dienstleistung erfolgt partnerschaftlich und ist auf eine langjährige Zusammenarbeit angelegt.

Märkte

- Branchenübergreifend
- Über 600 Kunden

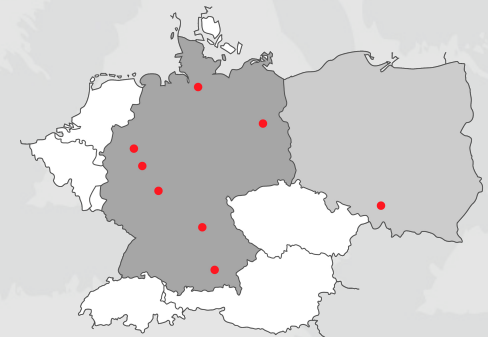


Leistungsangebot

- Business IT Alignment
- Business Information Management
- Business Process Management
- Anwendungsentwicklung
- SOA und System-Integration
- IT-Infrastruktur-Management

Eckdaten

- Gründung 1990
- 400 Mitarbeiter
- 8 Standorte



Wer sind Sie?



In diesem Vortrag geht's um ...

... automatisiertes Testen

... durch Entwickler

Warum testen?



Warum JavaScript-Code testen?



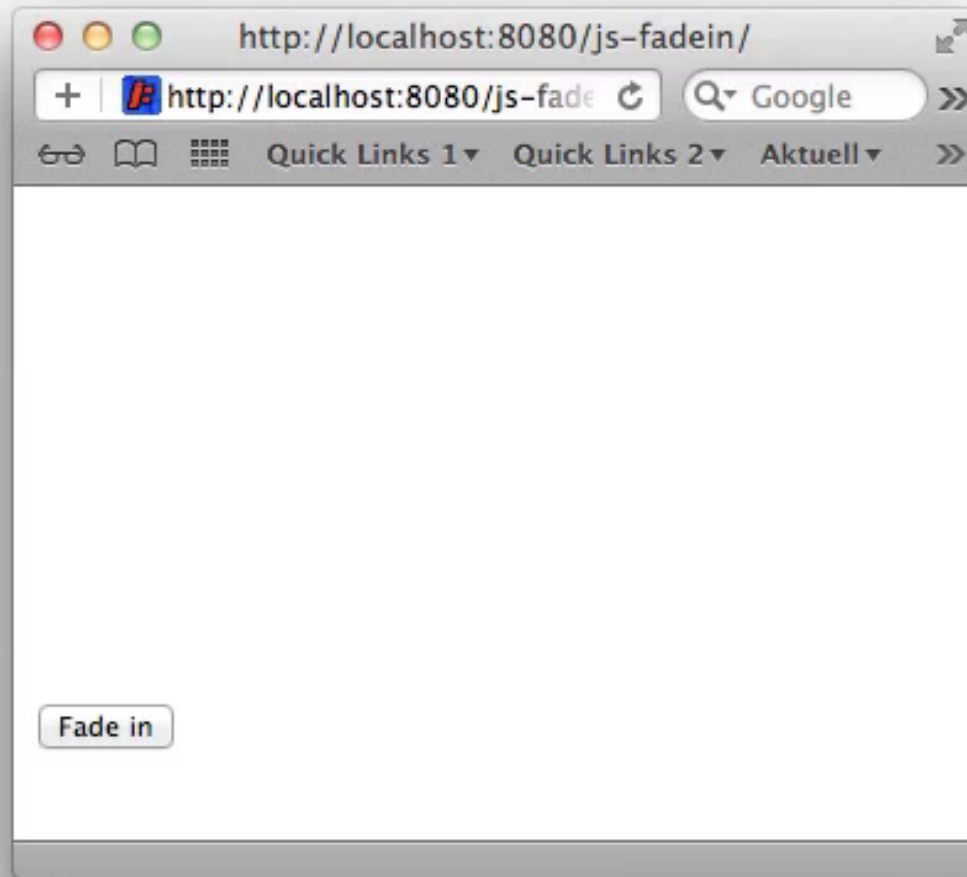
Laufzeitumgebungen für JavaScript



In diesem Vortrag:



Unser Beispiel



Tests formulieren



„Klassisch“ mit xUnit-Asserts

„BDD-Style“ à la RSpec

Demo: BDD mit Jasmine



Hello Jasmine!

```
describe("parseFloat", function() {  
  
    it("should return undefined for undefined", function () {  
        expect(util.parseFloat(undefined)).toBeUndefined();  
    });  
  
    it("should return 0 for empty string", function () {  
        expect(util.parseFloat('')).toEqual(0);  
    });  
  
    it("should return number for number strings", function () {  
        expect(util.parseFloat('1.5')).toEqual(1.5);  
    });  
  
    // ...  
  
});
```

Hello Jasmine!

```
describe("parseFloat", function() {
```

```
  it("should return undefined for undefined", function () {  
    expect(util.parseFloat(undefined)).toBeUndefined();  
  });
```

```
  it("should return 0 for empty string", function () {  
    expect(util.parseFloat('')).toEqual(0);  
  });
```

```
  it("should return number for number strings", function () {  
    expect(util.parseFloat('1.5')).toEqual(1.5);  
  });
```

```
  // ...
```

```
});
```



Suite

Hello Jasmine!

```
describe("parseFloat", function() {  
  
  it("should return undefined for undefined", function () {  
    expect(util.parseFloat(undefined)).toBeUndefined();  
  });  
  
  it("should return 0 for empty string", function () {  
    expect(util.parseFloat('')).toEqual(0);  
  });  
  
  it("should return number for number strings", function () {  
    expect(util.parseFloat('1.5')).toEqual(1.5);  
  });  
  
  // ...  
  
});
```



Hello Jasmine!

```
describe("parseFloat", function() {  
  
  it("should return undefined for undefined", function () {  
    expect(util.parseFloat(undefined)).toBeUndefined();  
  });  
  
  it("should return 0 for empty string", function () {  
    expect(util.parseFloat('')).toEqual(0);  
  });  
  
  it("should return number for number strings", function () {  
    expect(util.parseFloat('1.5')).toEqual(1.5);  
  });  
  
  // ...  
  
});
```

Matcher

Hello Jasmine!

Matcher

```
expect(x).toEqual(y);  
expect(x).toBe(y);  
expect(x).toBeDefined();  
expect(x).toBeUndefined();  
expect(x).toBeTruthy();  
expect(x).toBeFalsy();  
...
```

Hello Jasmine!

Spec Runner

jasmine 1.0.2 revision 1298837858 Show passed skipped

4 specs, 0 failures in 0.004s Finished at Fri Sep 02 2011 07:16:03 GMT+0200 (CEST) run all

parseFloat	run
<u>should return undefined for undefined</u>	run
<u>should return 0 for empty string</u>	run
<u>should return the number for number strings</u>	run
<u>should return the number from a string with numbers and characters</u>	run

Fortgeschrittene Konzepte



Spies



Spies

```
describe("execute fadein", function() {
  it("should eventually set opacity to 1", function () {
    var element = document.createElement("div");
    jasmine.Clock.useMock();
    spyOn(util, 'opacity');
    fadein.execute(element);
    jasmine.Clock.tick(1000);
    var mostRecentCall = util.opacity.mostRecentCall;
    expect(mostRecentCall.args[0]).toBe(element);
    expect(mostRecentCall.args[1]).toEqual(1);
  });
});
```

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function () {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(1000);  
    var mostRecentCall = util.opacity.mostRecentCall;  
    expect(mostRecentCall.args[0]).toBe(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(1000);  
    var mostRecentCall = util.opacity.mostRecentCall;  
    expect(mostRecentCall.args[0]).toBe(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```

Arrange

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(1000);  
    var mostRecentCall = util.opacity.mostRecentCall;  
    expect(mostRecentCall.args[0]).toBe(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```

Arrange

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(100);  
    var mostRecentCall = util.opacity.calls.mostRecent();  
    expect(mostRecentCall.args[0]).toEqual(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```

Arrange

```
// set opacity of 'element' to 'opacity'  
util.opacity = function (element, opacity) {  
  ...  
};
```

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function () {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(1000);  
    var mostRecentCall = util.opacity.mostRecentCall;  
    expect(mostRecentCall.args[0]).toBe(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```



Act

Spies

```
describe("execute fadein", function() {  
  it("should eventually set opacity to 1", function () {  
    var element = document.createElement("div");  
    jasmine.Clock.useMock();  
    spyOn(util, 'opacity');  
    fadein.execute(element);  
    jasmine.Clock.tick(1000);  
    var mostRecentCall = util.opacity.mostRecentCall;  
    expect(mostRecentCall.args[0]).toBe(element);  
    expect(mostRecentCall.args[1]).toEqual(1);  
  });  
});
```



Act

Spies

```
describe("execute fadein", function() {
  it("should eventually set opacity to 1", function () {
    var element = document.createElement("div");
    jasmine.Clock.useMock();
    spyOn(util, 'opacity');
    fadein.execute(element);
    jasmine.Clock.tick(1000);
    var mostRecentCall = util.opacity.mostRecentCall;
    expect(mostRecentCall.args[0]).toBe(element);
    expect(mostRecentCall.args[1]).toEqual(1);
  });
});
```

Assert

Spies

```
describe("execute", function() {
  it("should call the function", function() {
    var element = document.createElement("div");
    jasmine.clock().install();
    spyOn(util, "opacity");
    fadein.execute(element);
    jasmine.clock().tick(1000);
    var mostRecentCall = util.opacity.mostRecentCall;
    expect(mostRecentCall.args[0]).toBe(element);
    expect(mostRecentCall.args[1]).toEqual(1);
  });
});
```

```
// set opacity of 'element' to 'opacity'
util.opacity = function (element, opacity) {
  ...
};
```

Assert

Funktion	Der Spy soll ...
<code>spy.andCallThrough()</code>	... die ursprüngliche Funktion aufrufen
<code>spy.andReturn(argument)</code>	... das übergebene Argumente zurückgeben
<code>spy.andThrow(exception)</code>	... die übergebene Exception werfen
<code>spy.andCallFake(function)</code>	... die übergebene Funktion aufrufen

Funktionen von Spies

Funktion	Prüft, ob der Spy ...
<code>toHaveBeenCalled()</code>	... aufrufen wurde
<code>toHaveBeenCalledWith(args)</code>	... mit den übergebenen Argumenten aufgerufen wurde

Matcher für Spies

Funktion	Bedeutung
<code>spy.callCount</code>	Anzahl der Aufrufe
<code>spy.argsForCall[i]</code>	Argumente des i-ten Aufrufs
<code>spy.mostRecentCall.args</code>	Argumente des letzten Aufrufs

Eigenschaften von Spies

Asynchrone Tests



Asynchrone Tests

```
it("should eventually set opacity to 1", function() {
    var element = document.createElement("div");
    spyOn(util, 'opacity');
    runs(function() {
        fadein.execute(element);
    });
    waitsFor(function () {
        return opacitySpy.mostRecentCall.args[1] === 1;
    });
    runs(function() {
        // some other expectations
        expect(opacitySpy.mostRecentCall.args[1]).toEqual(1);
    });
});
```

Asynchrone Tests

```
it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    spyOn(util, 'opacity');  
    runs(function() {  
        fadein.execute(element);  
    });  
    waitsFor(function () {  
        return opacitySpy.mostRecentCall.args[1] === 1;  
    });  
    runs(function() {  
        // some other expectations  
        expect(opacitySpy.mostRecentCall.args[1]).toEqual(1);  
    });  
});
```

Arrange

Asynchrone Tests

```
it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    spyOn(util, 'opacity');  
    runs(function() {  
        fadein.execute(element);  
    });  
    waitsFor(function () {  
        return opacitySpy.mostRecentCall.args[1] === 1;  
    });  
    runs(function() {  
        // some other expectations  
        expect(opacitySpy.mostRecentCall.args[1]).toEqual(1);  
    });  
});
```



Act

Asynchrone Tests

```
it("should eventually set opacity to 1", function() {
  var element = document.createElement("div");
  spyOn(UTIL, 'opacity');
  runs(function() {
    fadein.execute(element);
  });
  waitsFor(function () {
    return opacitySpy.mostRecentCall.args[1] === 1;
  });
  runs(function() {
    // some code
    expect(
  });
});
```

Wait...

```
// set opacity of 'element' to 'opacity'
UTIL.opacity = function (element, opacity) {
  ...
};
```


Asynchrone Tests

```
it("should eventually set opacity to 1", function() {  
    var element = document.createElement("div");  
    spyOn(util, 'opacity');  
    runs(function() {  
        fadein.execute(element);  
    });  
    waitsFor(function () {  
        return opacitySpy.mostRecentCall.args[1] === 1;  
    });  
    runs(function() {  
        // some other expectations  
        expect(opacitySpy.mostRecentCall.args[1]).toEqual(1);  
    });  
});
```

Assert

UI-Tests

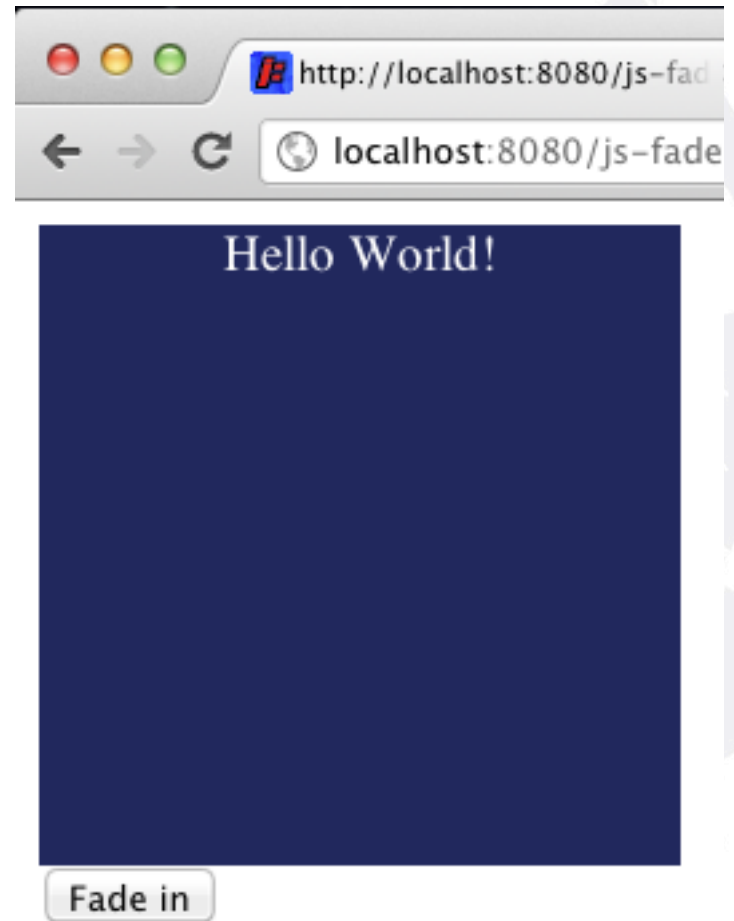


```
<html>
<body>

<div id="hello">Hello World!</div>

<button id="fadein">Fade in</button>

</body>
</html>
```



```
describeUi("index", "/js-fadein/index.html", function () {  
  
    it("should fade in hello div on button click", function () {  
        var win, field, button;  
        runs(function() {  
            field = document.getElementById('hello');  
            button = document.getElementById('fadein');  
            expect(util.opacity(field)).toEqual(0);  
            simulate(button, 'click');  
        });  
        runs(function () {  
            expect(util.opacity(field)).toEqual(1);  
        });  
    });  
  
});
```

```
describeUi("index", "/js-fadein/index.html", function () {  
  
  it("should fade in hello div on button click", function () {  
    var win, field, button;  
    runs(function() {  
      field = document.getElementById('hello');  
      button = document.getElementById('fadein');  
      expect(util.opacity(field)).toEqual(0);  
      simulate(button, 'click');  
    });  
    runs(function () {  
      expect(util.opacity(field)).toEqual(1);  
    });  
  });  
});  
  
});
```

Page to load

```
describeUi("index", "/js-fadein/index.html", function () {
```

```
    it("should fade in hello div on button click", function () {
        var win, field, button;
        runs(function() {
            field = document.getElementById('hello');
            button = document.getElementById('fadein');
            expect(util.opacity(field)).toEqual(0);
            simulate(button, 'click');
        });
        runs(function () {
            expect(util.opacity(field)).toEqual(1);
        });
    });
});
```

```
describeUi("index", "/js-fadein/index.html", function () {  
  it("should fade in hello div on button click", function () {  
    var win, field, button;  
    runs(function() {  
      field = document.getElementById('hello');  
      button = document.getElementById('fadein');  
      expect(util.opacity(field)).toEqual(0);  
      simulate(button, 'click');  
    });  
    runs(function () {  
      expect(util.opacity(field)).toEqual(1);  
    });  
  });  
});
```

Precondition

```
describeUi("index", "/js-fadein/index.html", function () {  
  
  it("should fade in hello div on button click", function () {  
    var win, field, button;  
    runs(function() {  
      field = document.getElementById('hello');  
      button = document.getElementById('fadein');  
      expect(util.opacity(field)).toEqual(0);  
      simulate(button, 'click');  
    });  
    runs(function () {  
      expect(util.opacity(field)).toEqual(1);  
    });  
  });  
  
});
```

Act


```
describeUi("index", "/js-fadein/index.html", function () {  
  
    it("should fade in hello div on button click", function () {  
        var win, field, button;  
        runs(function() {  
            field = document.getElementById('hello');  
            button = document.getElementById('fadein');  
            expect(util.opacity(field)).toEqual(0);  
            simulate(button, 'click');  
        });  
        runs(function () {  
            expect(util.opacity(field)).toEqual(1);  
        });  
    });  
  
});
```

Assert

Testausführung automatisieren

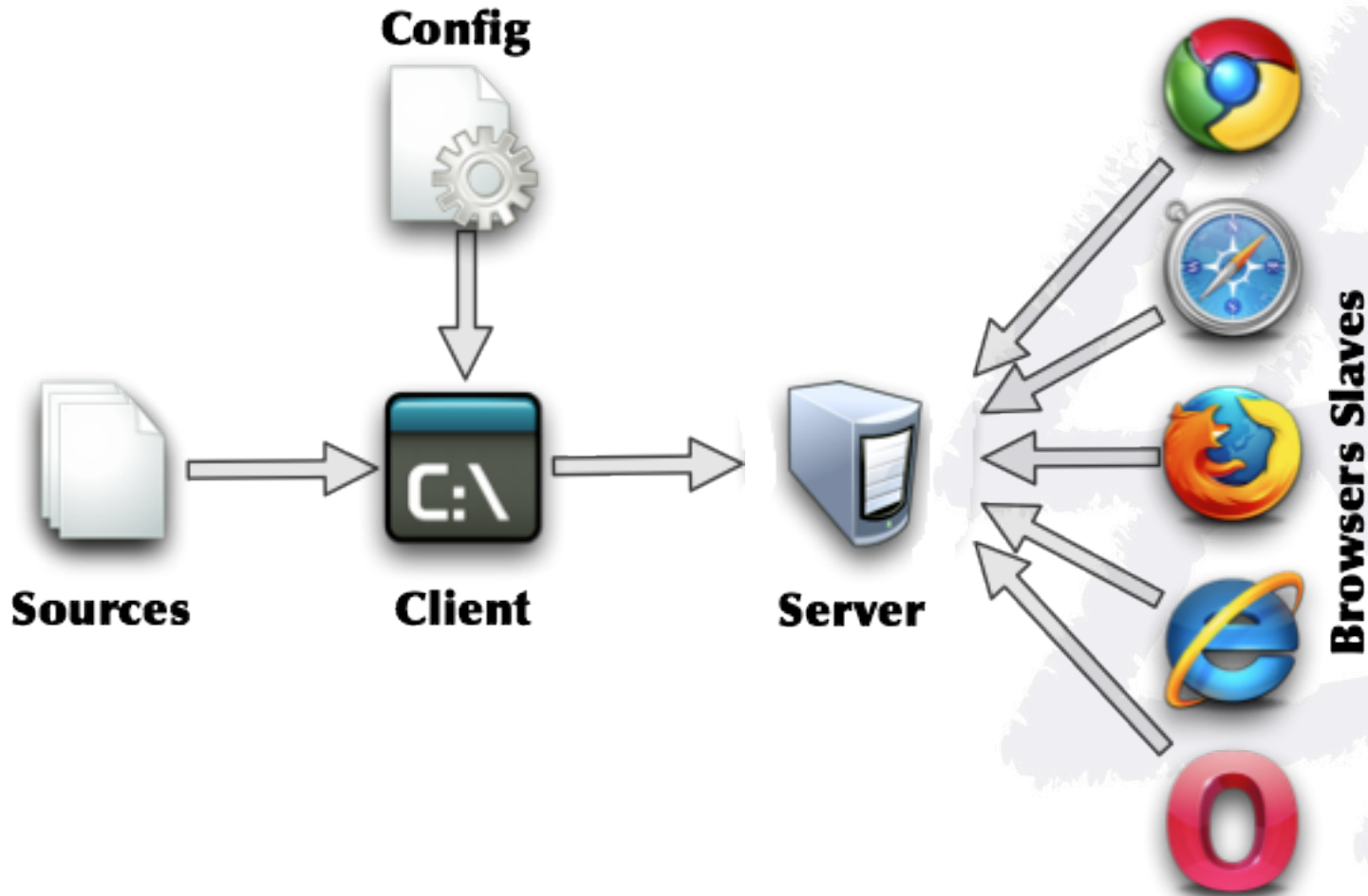


	Pro	Contra	Beispiele
Im Browser testen	Laufen in Produktionsumgebung	- Schwerer zu automatisieren - Schwerer in CI einzubetten	QUnit YUI Test Jasmine
Headless Browser	- Leicht zu automatisieren - Leicht in CI einzubetten	Browser wird nur simuliert	Env ^{js} PhantomJS Zombie.js HtmlUnit

Browser fernsteuern



JsTestDriver



<http://code.google.com/p/js-test-driver/>

Demo: JsTestDriver



Fazit

**Es gibt ein umfangreiches Toolset zum Testen
von JavaScript-Code.**

Testen Sie Ihren Code!

Testen Sie insbesondere Ihren JavaScript-Code!

Sie haben keine Ausrede ...

Links

Jasmine	http://pivotal.github.com/jasmine/
Jasmine UI	https://github.com/tigbro/jasmine-ui
js-fadein	https://github.com/stefanscheidt/js-fadein
JsTestDriver	http://code.google.com/p/js-test-driver/

saturated writing

(By Eduordo, <http://www.flickr.com/photos/tnarik/366393127/>)

Smiley Keyboard

(By ~Prescott, <http://www.flickr.com/photos/ppym1/93571524/>)

Spyhole, Paris

(By smith of smiths, <http://www.flickr.com/photos/hesketh/232015302/>)

Sorta synchronised diving

(By Aaron Retz, <http://www.flickr.com/photos/aretz/309469339/>)

Stamp Collector

(By C. Castillo, <http://www.flickr.com/photos/carolinedecastillo/2750577684/>)

bios [bible]

(By Gastev, <http://www.flickr.com/photos/gastev/2174505811/>)

Day 276/365: in hot pursuit

(By Tina Vega, <http://www.flickr.com/photos/tinavega/3066602429/>)

**Vielen Dank
für Ihr Interesse!**

@tigbro



OPITZ CONSULTING