Das *lustige* *Überlebenshandbuch* *für* JavaScript

Oliver Pehnke, Benjamin Schmid

exXcellent solutions

JavaScript

ist die **erfolgreichste** Sprache

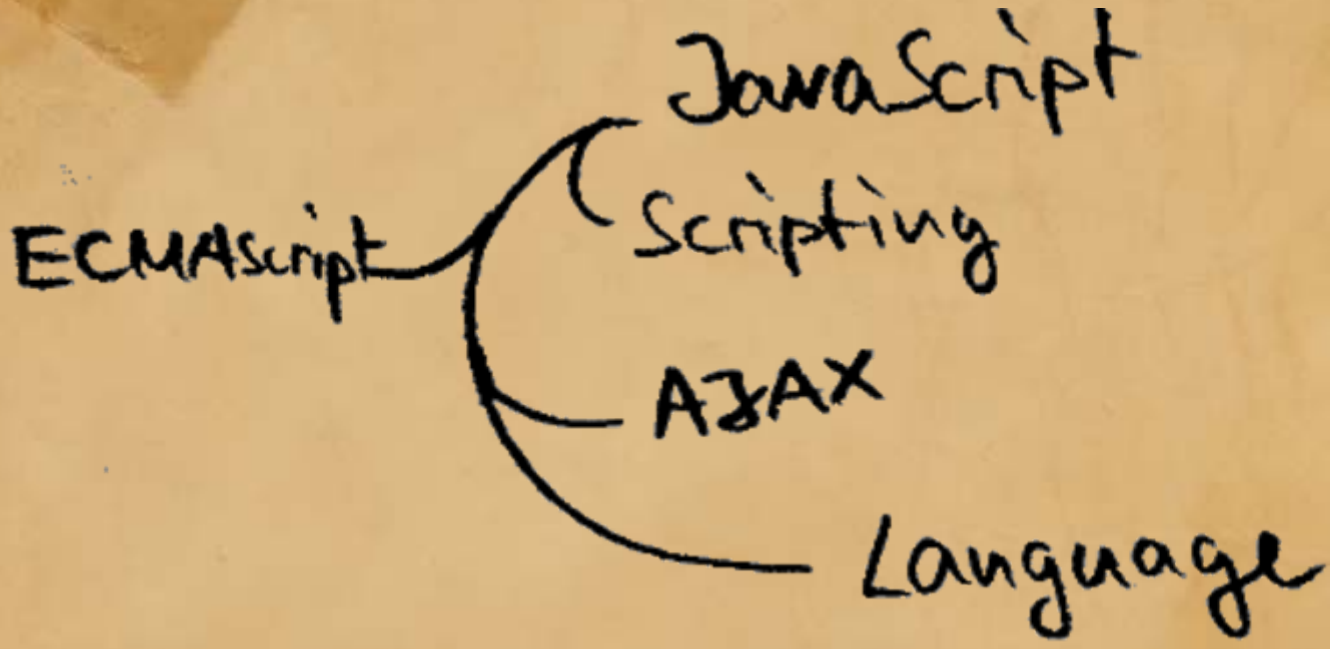
aller Zeiten

im bekannten **Universum**.

Inhaltsverzeichnis

Prolog

Evolution von Javascript



Ein erster Vergleich

Java

- Klassen-basiert
- statisch und stark typisiert

JavaScript

- Prototype-basiert
- dynamisch und schwach typisiert

JavaScripts Griff nach den Sternen

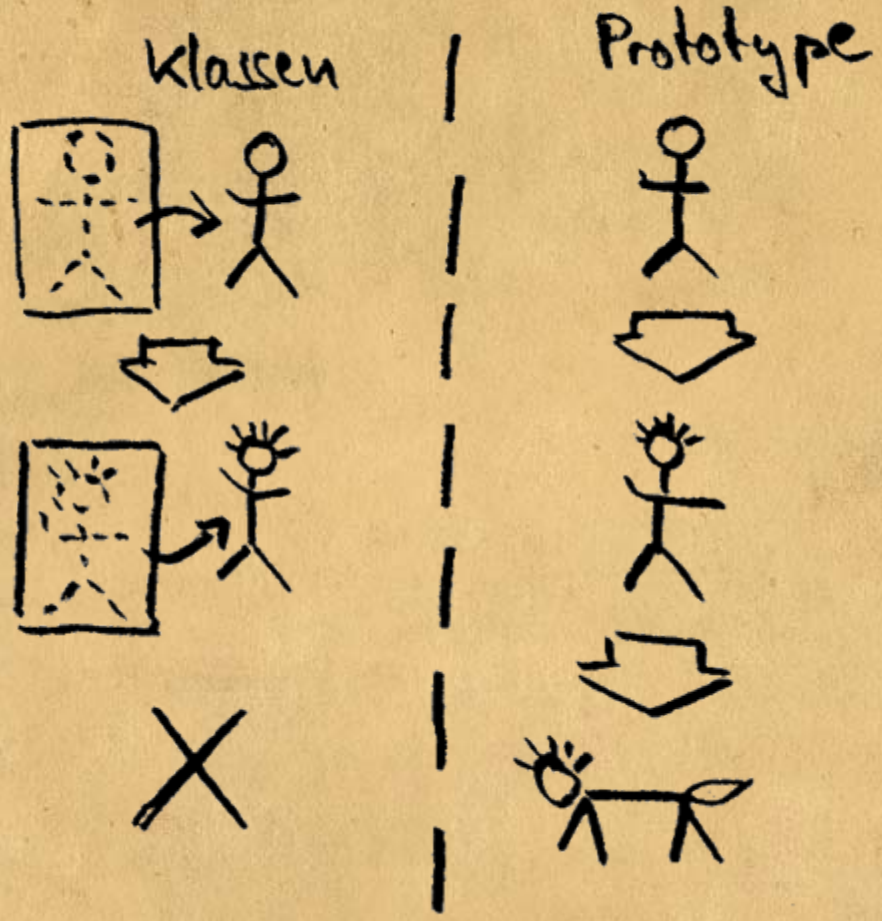
JavaScript nur im Browser? Weit gefehlt!

- *Desktop:* Widgets, Gnome3 Shell
- *Backend*: Couch DB
- *Server*: NodeJS, RingoJS, Jaxer (Server Side JS)

Akt I Sprache

Der Kern von JavaScript

Akt I



- Prototypische ObjektOrientierung: **(klassenlos)**
- **First class-Function**
- **Function** Scope

Beispiel klassenlose OO: String.trim() nachrüsten

Akt I

```
if (typeof String.prototype.trim !== 'function') {
  String.prototype.trim = function () {
      return this.replace(
          /^\s*(\S*(\s+\S+)*)\s*$/, „$1“);
  };
}
```

Beispiel klassenlose OO: String.trim() nachrüsten

Akt I

```
if (typeof String.prototype.trim !== 'function') {
  String.prototype.trim = function () {
    return this.replace(
        /^\s+|\s+$/g, '');
  };
}
```

Die Schlacht um die Vorherrschaft hat begonnen!!!

Datentypen

Akt I

Datentyp	Java	Javascript
Text	String, Char	**String** (16 Bit Unicode)
Zahl	Double, Integer, Float, Byte, Long, Short	**number** (64bit floating point)**, NaN**
Ja/Nein	Boolean	**boolean**
Nix	null	**undefined, null**
[] (array)	Collection, Array[]	**Array , Object**
Funktion	-	**function**

Beispiel First Class Function


```
function Mix( ingredient1, ingredient2, fx ) {
    print("Get the " + ingredient1);
    fx(ingredient1);
    fx(ingredient2);
}
```

Closures

Akt I

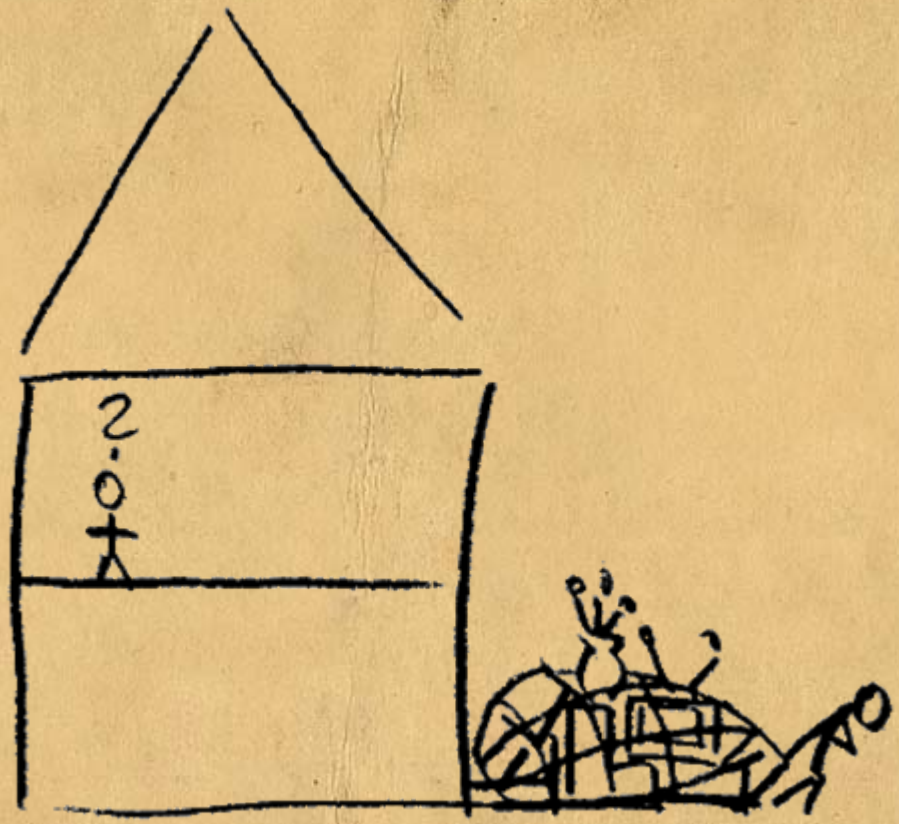
Closure = Wenn eine Function sich **merkt**, was um sie herum passiert.

Closures

Akt I

Closures

Akt I



Prototypische OO & Closures

```
var Person = {
        name: "Tim",
        greeting: function () {
                return "Hello " + this.name + ". ";
        }
};
Person.greeting();
```

Prototypische OO & Closures

```
var Person = {
        name: "Tim",
        greeting: function () {
                return "Hello " + this.name + ". ";
        }
};
Person.greeting();
```

{Hello Tim.}

Prototypische OO & Closures

```
var greeting = Person.greeting();
greeting(); //this => globales object
```

{Hello undefined}

This and That

Akt I

In **Javascript** ist der Wert von „**this**“ abhängig von der Art des Aufrufs.

Aufruf	this
function	global object undefined
method	object (Eigner der Methode)
constructor	Das „new“ object
apply	argument

Prototypische OO & Closures

```
var Dog = {
        name: “Struppi",
        greeting: Person.greeting()
};
Dog.greeting();
```

{Hello Struppi.}

Und was ist mit OO?

Akt I

```
function Container(param) {
    function dec() {
        if (secret > 0) {
            secret -= 1;
            return true;
        } else { return false; }
    }
    var secret = 3;
    var that = this;
    this.service = function () {
        if (dec()) {
            return that.secret;
        } else { return null; }
    };
}
```

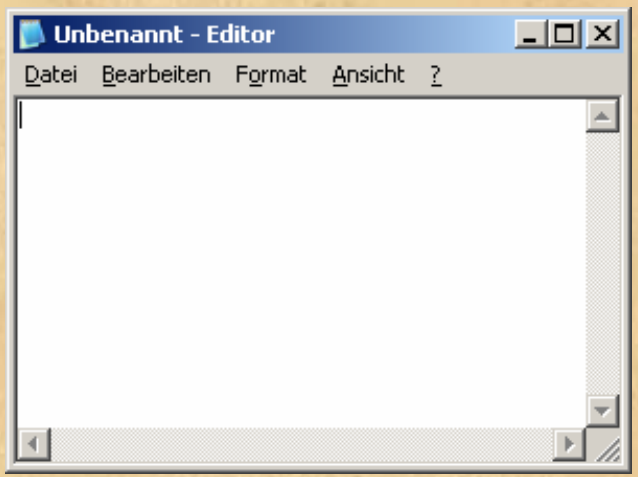
private

privileged

Akt II Werkzeuge

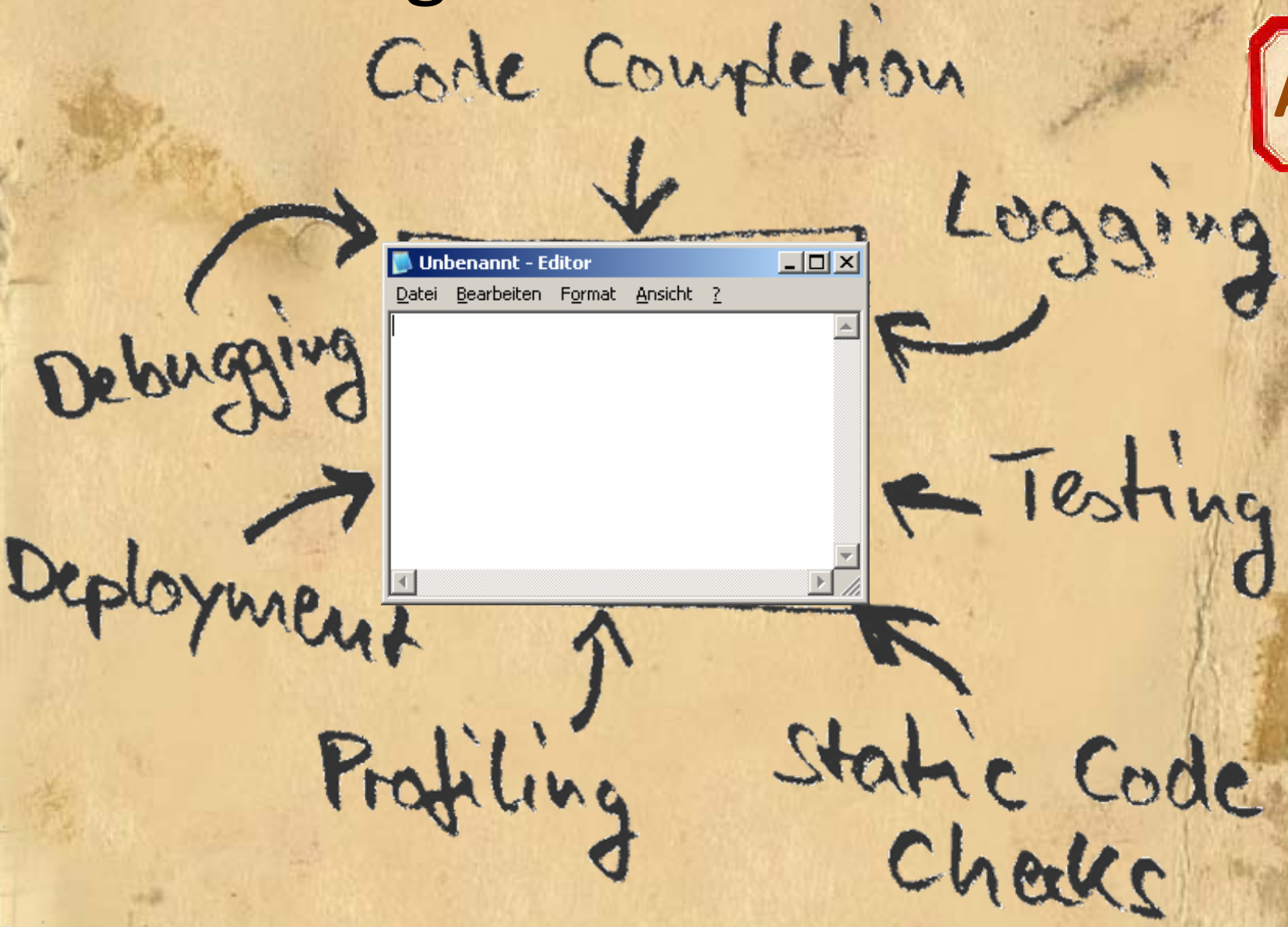
Werkzeuge: Was ist zu erwarten?

Akt II



Werkzeuge: Was ist zu erwarten?

Akt II



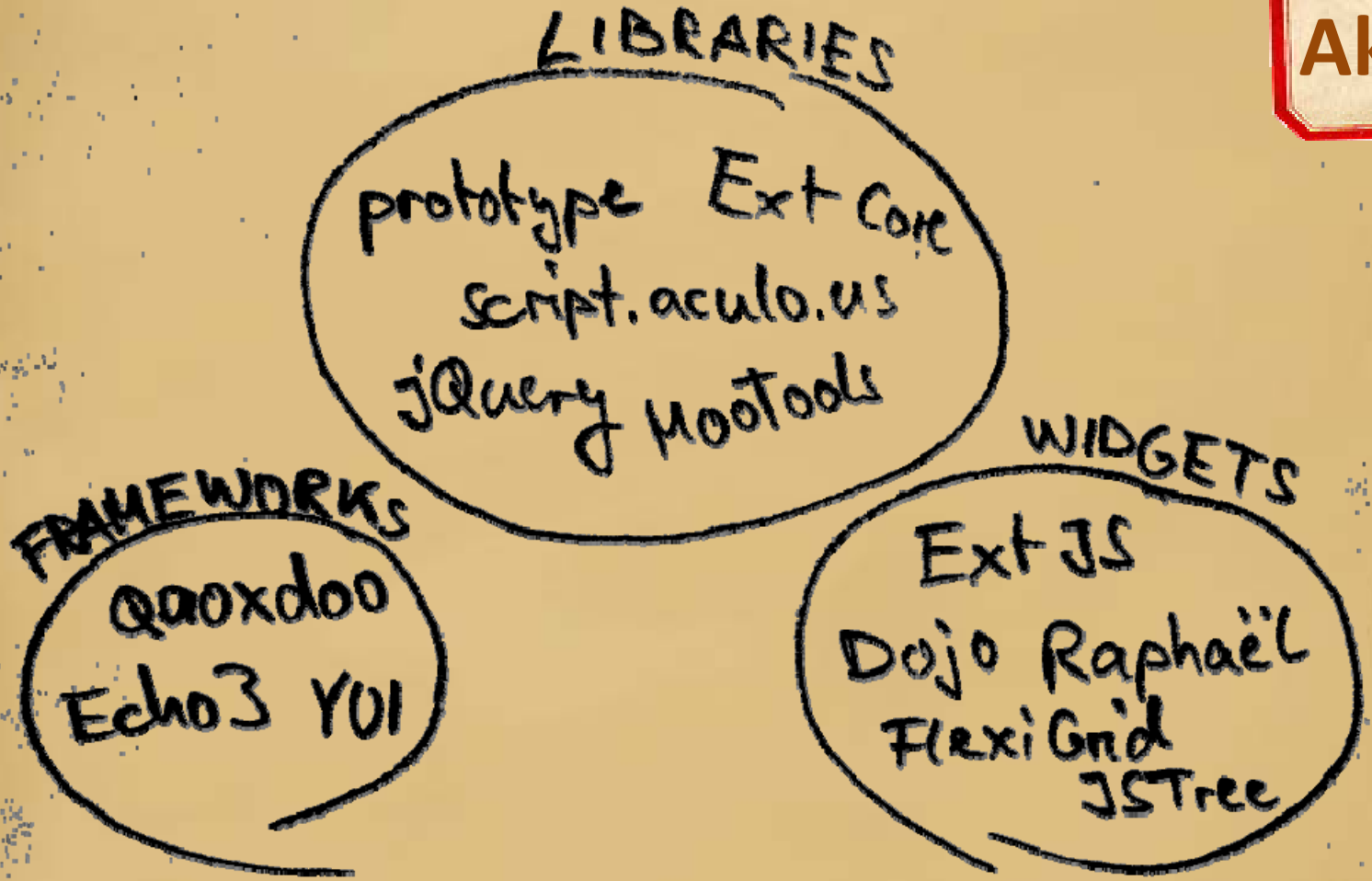
Die Küche empfiehlt:

Akt II

- **IDE & Umgebung**:
  - IntelliJ & Mozilla/Chrome
- **Debugging**:
  - IntelliJ, FireBug
- **Profiling**:
  - Firebug & Chrome
- **Logging**
  - window.console.log + FireBug

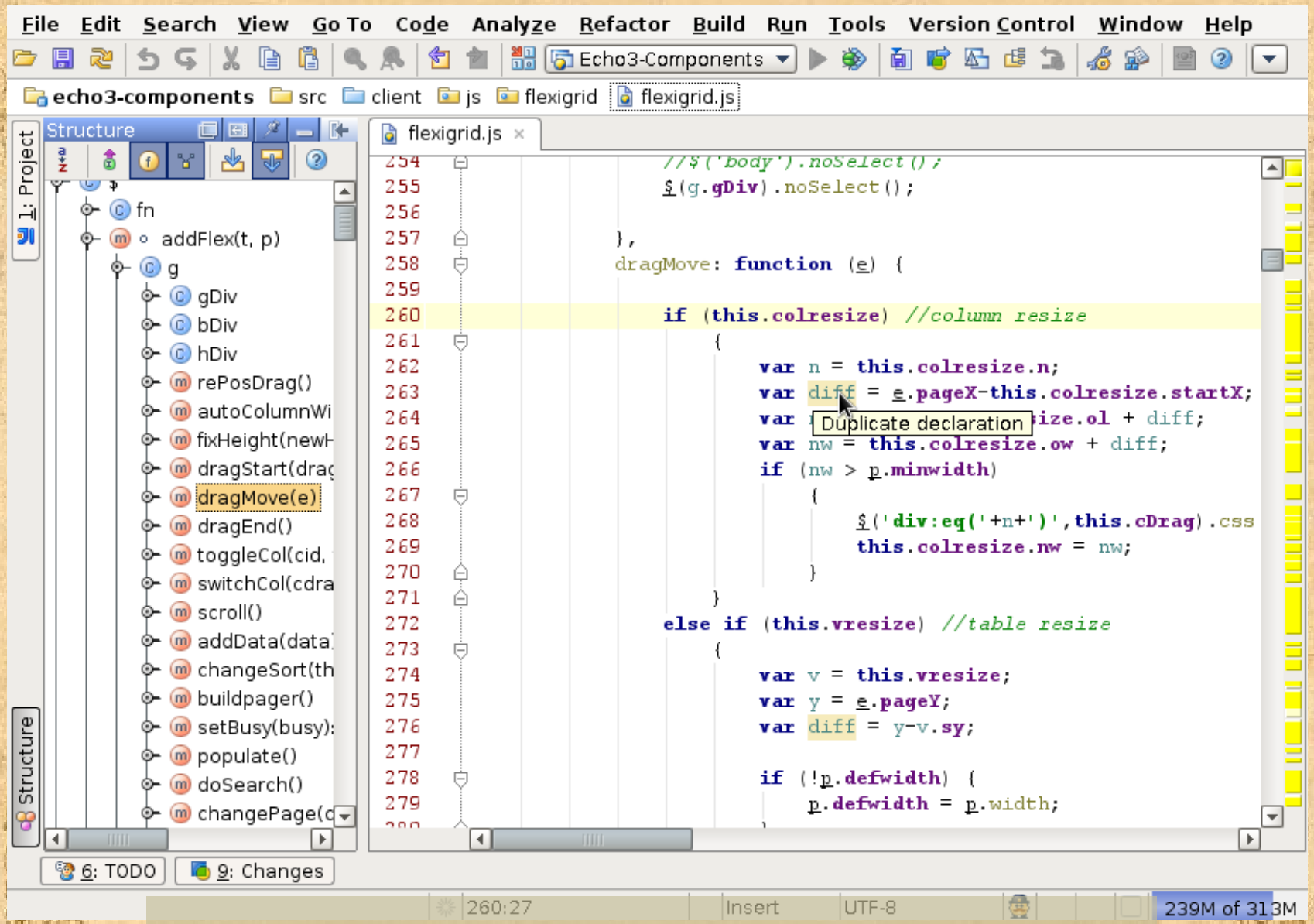
Die Wahrheit über Libraries

Akt II

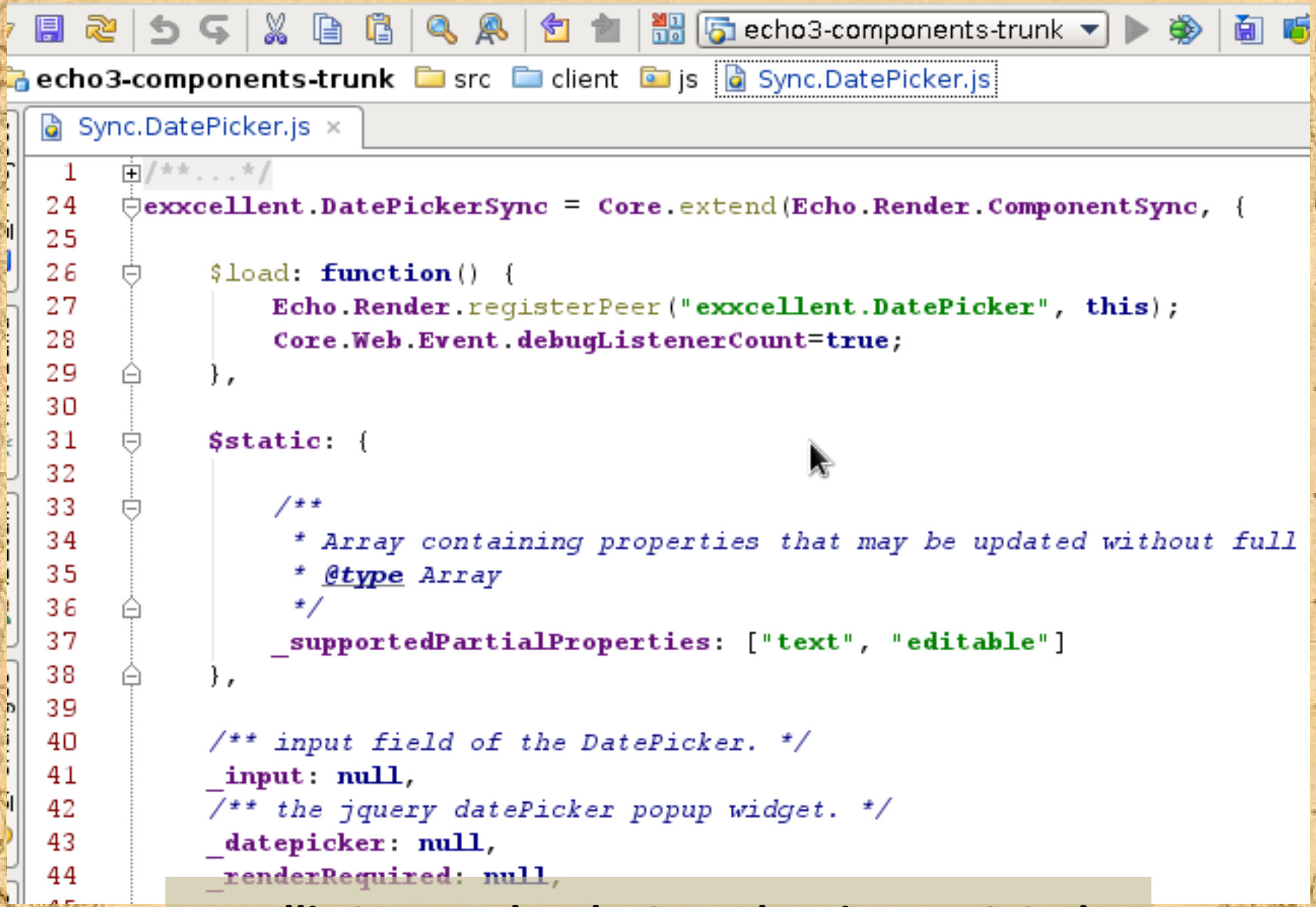


Akt II

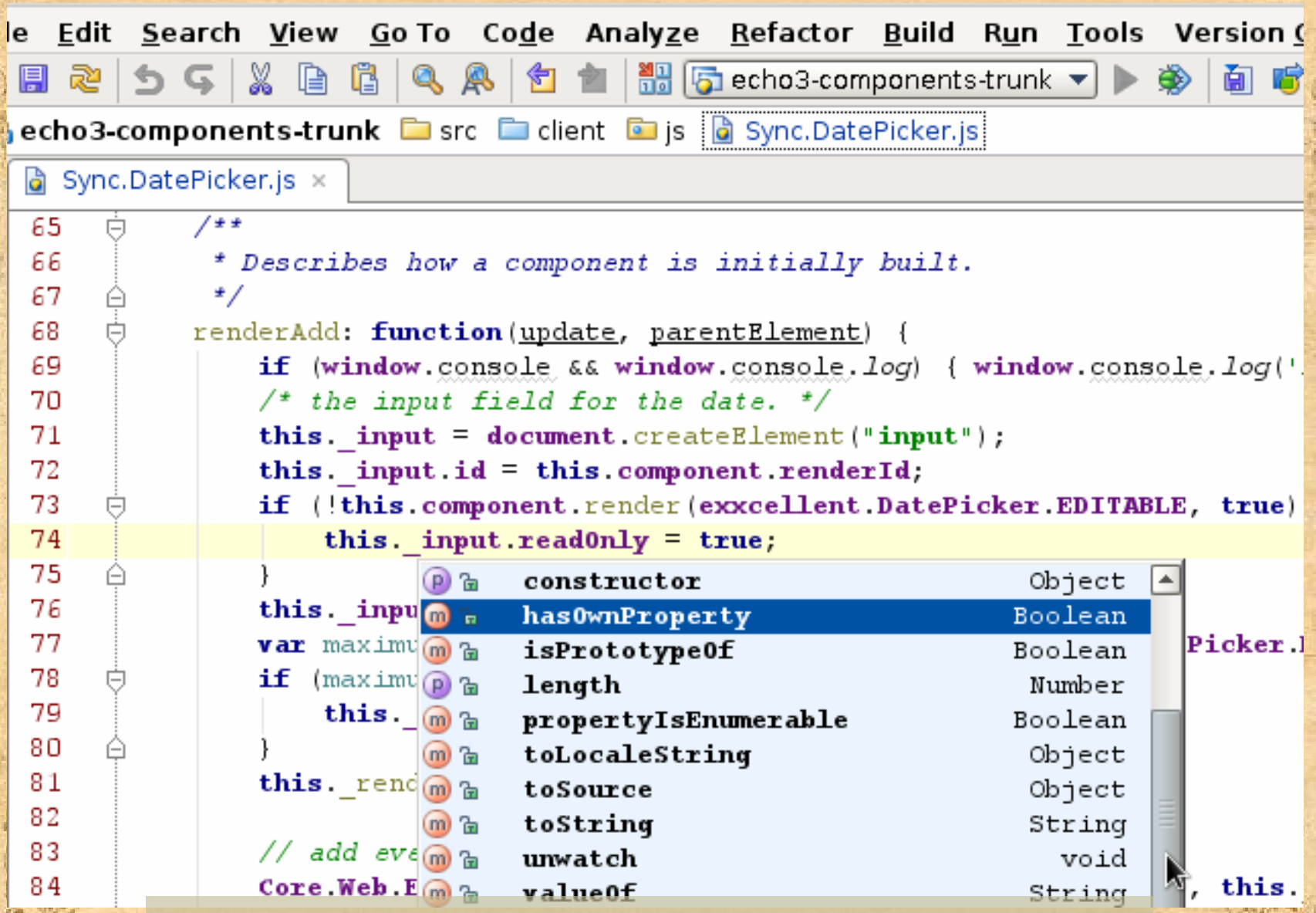
Demo



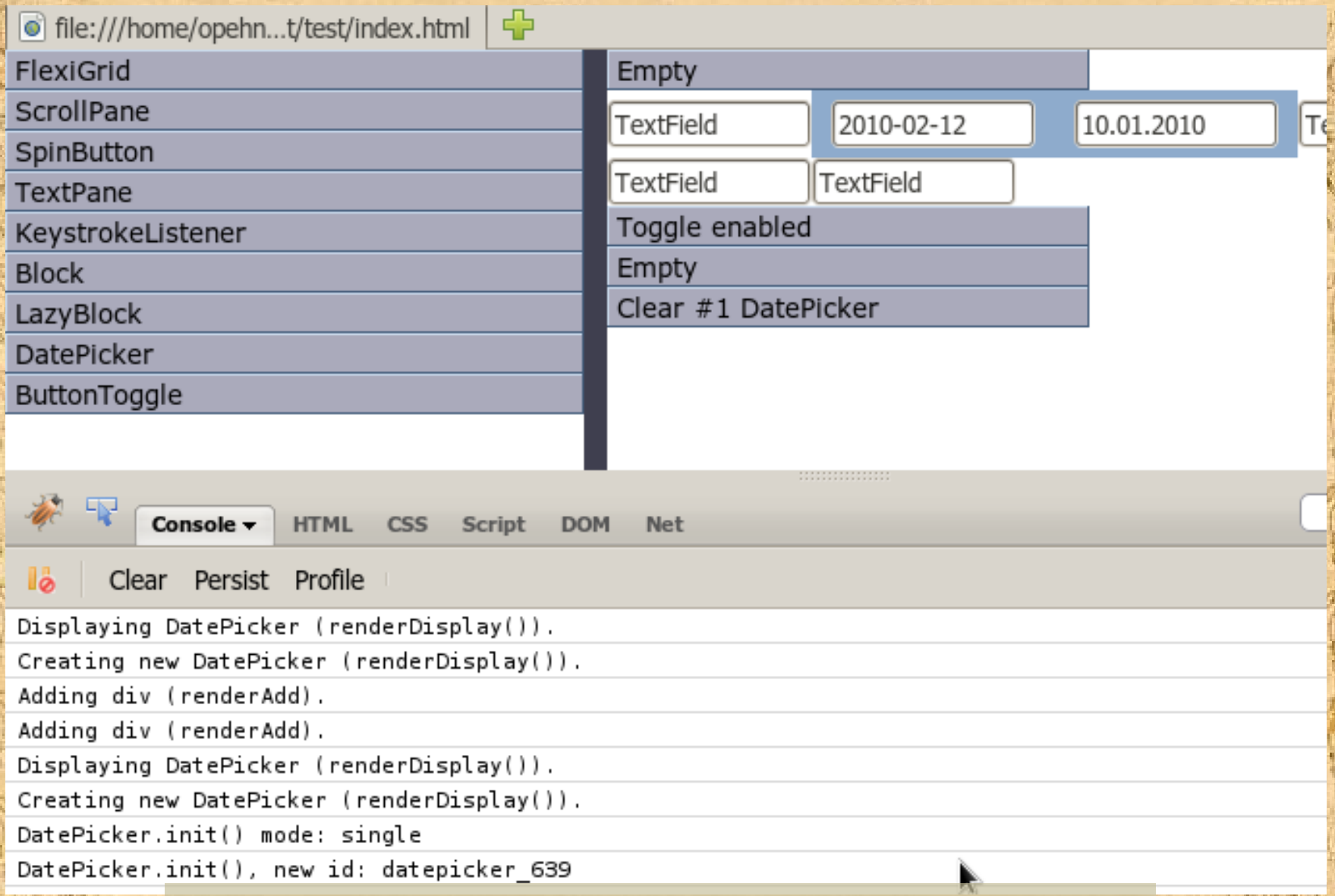
IntelliJ 9 – Codestruktur, Inspections



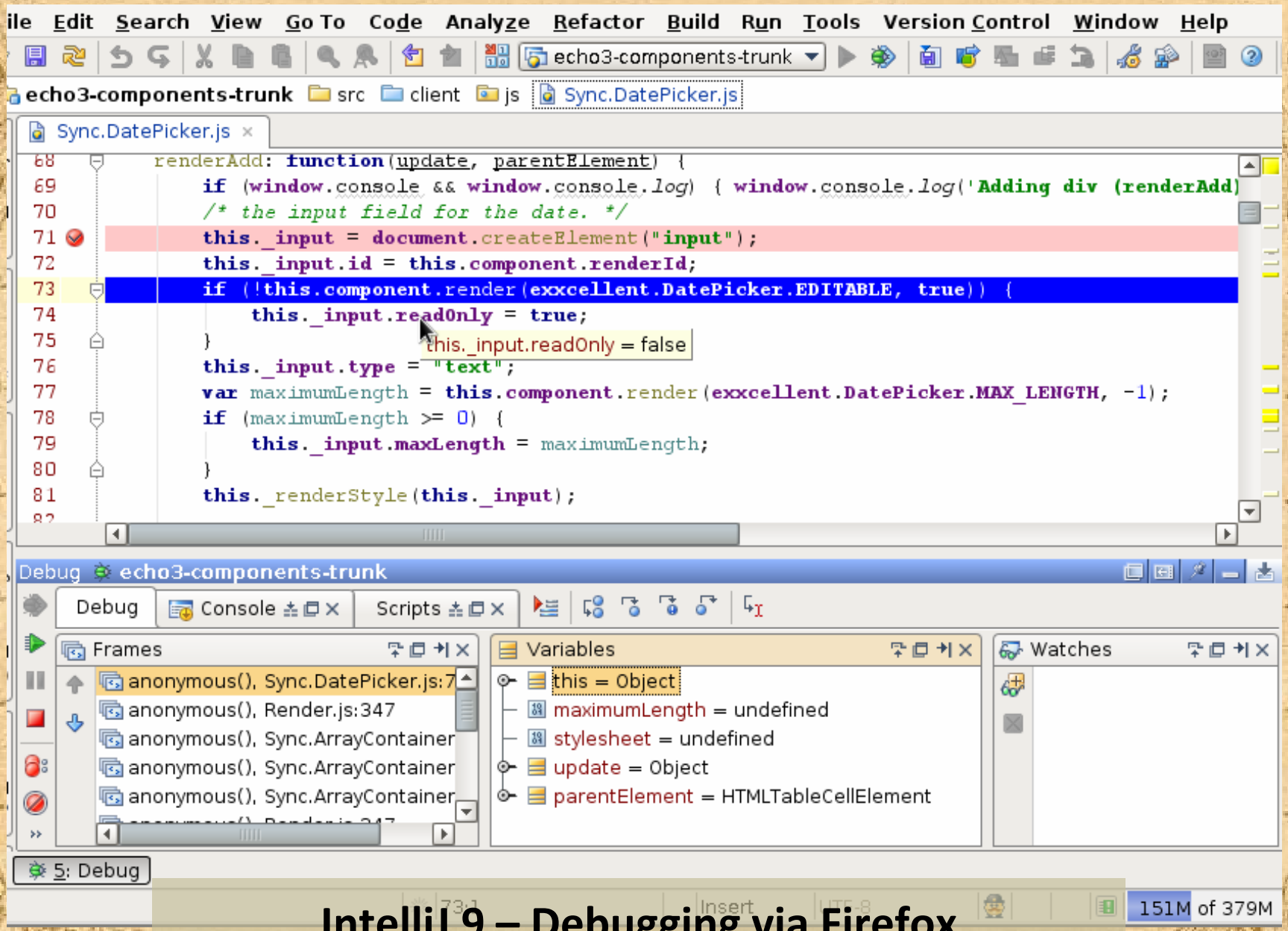
IntelliJ 9 – Durch Echo3 strukturierter JS Code



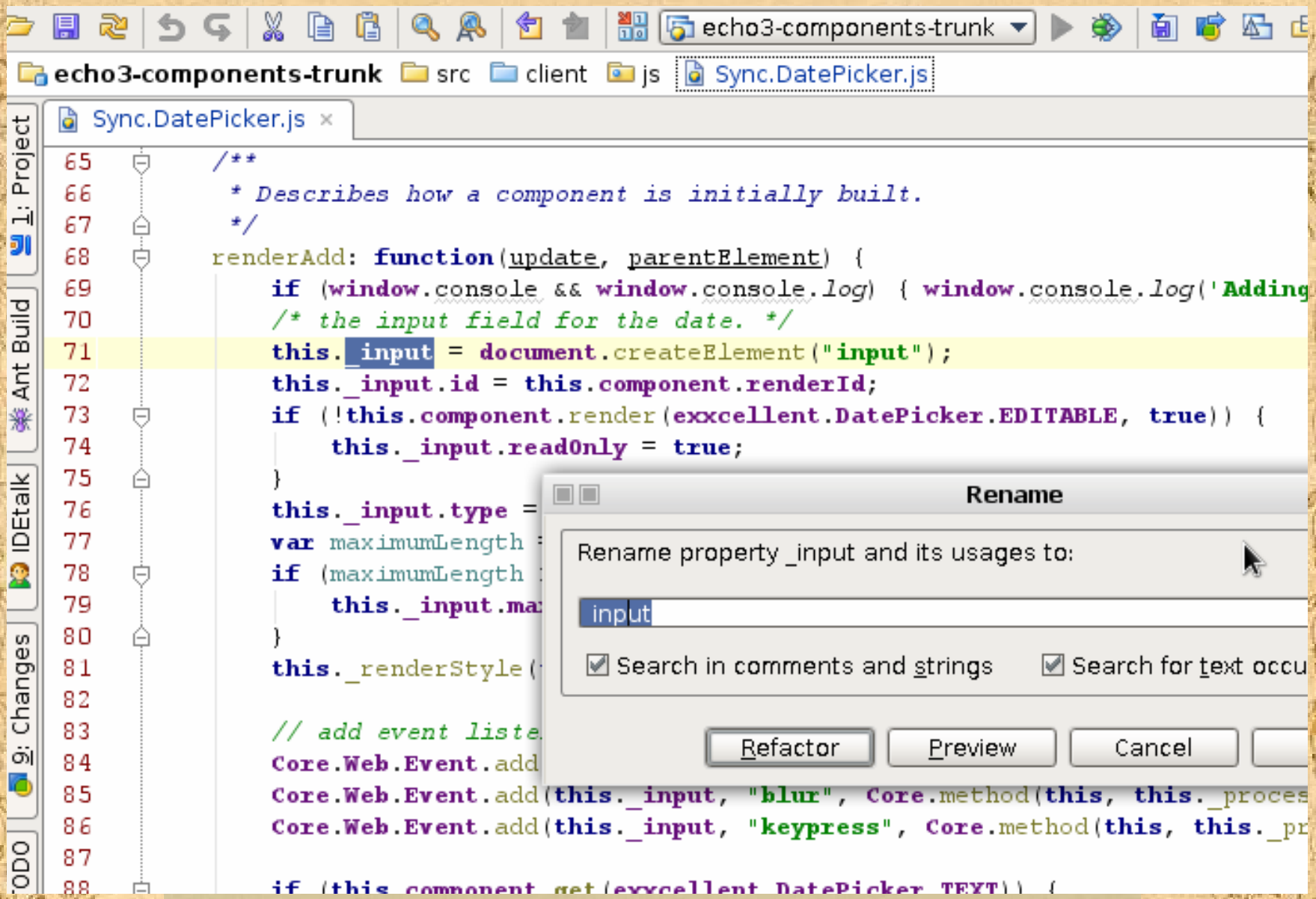
IntelliJ 9 – Best Guess Code Completition



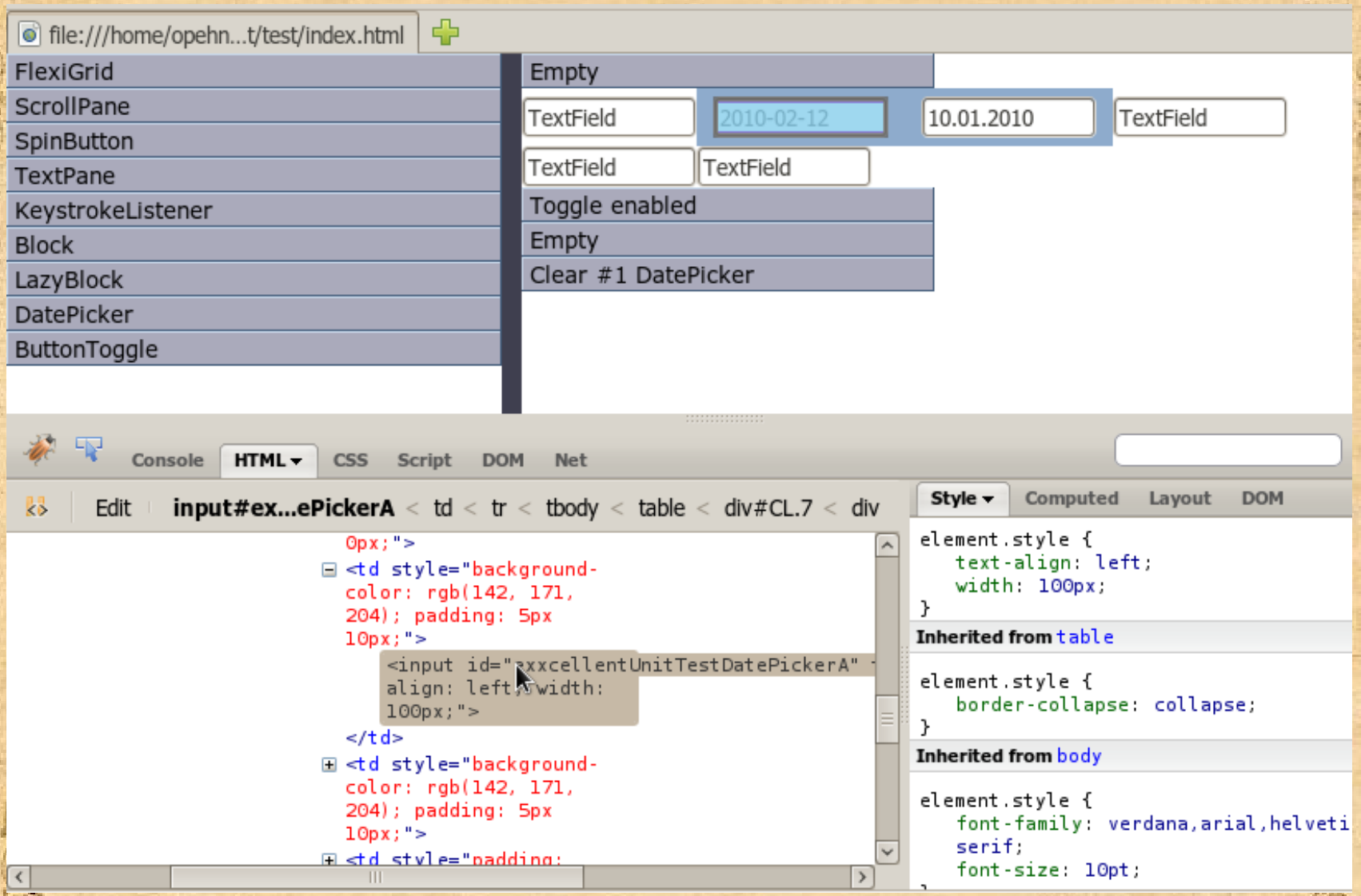
Firefox (Firebug) - Console



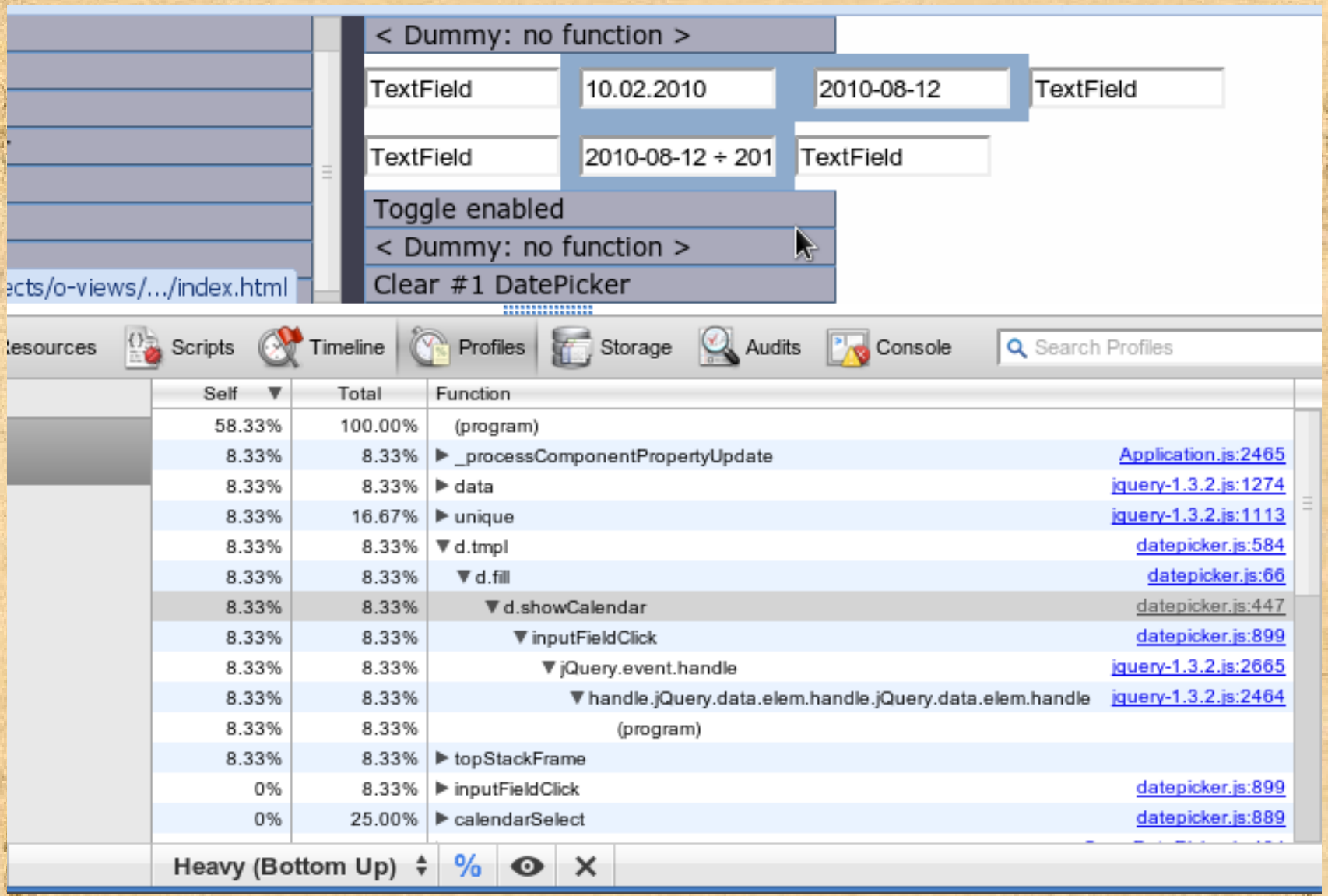
IntelliJ 9 – Debugging via Firefox



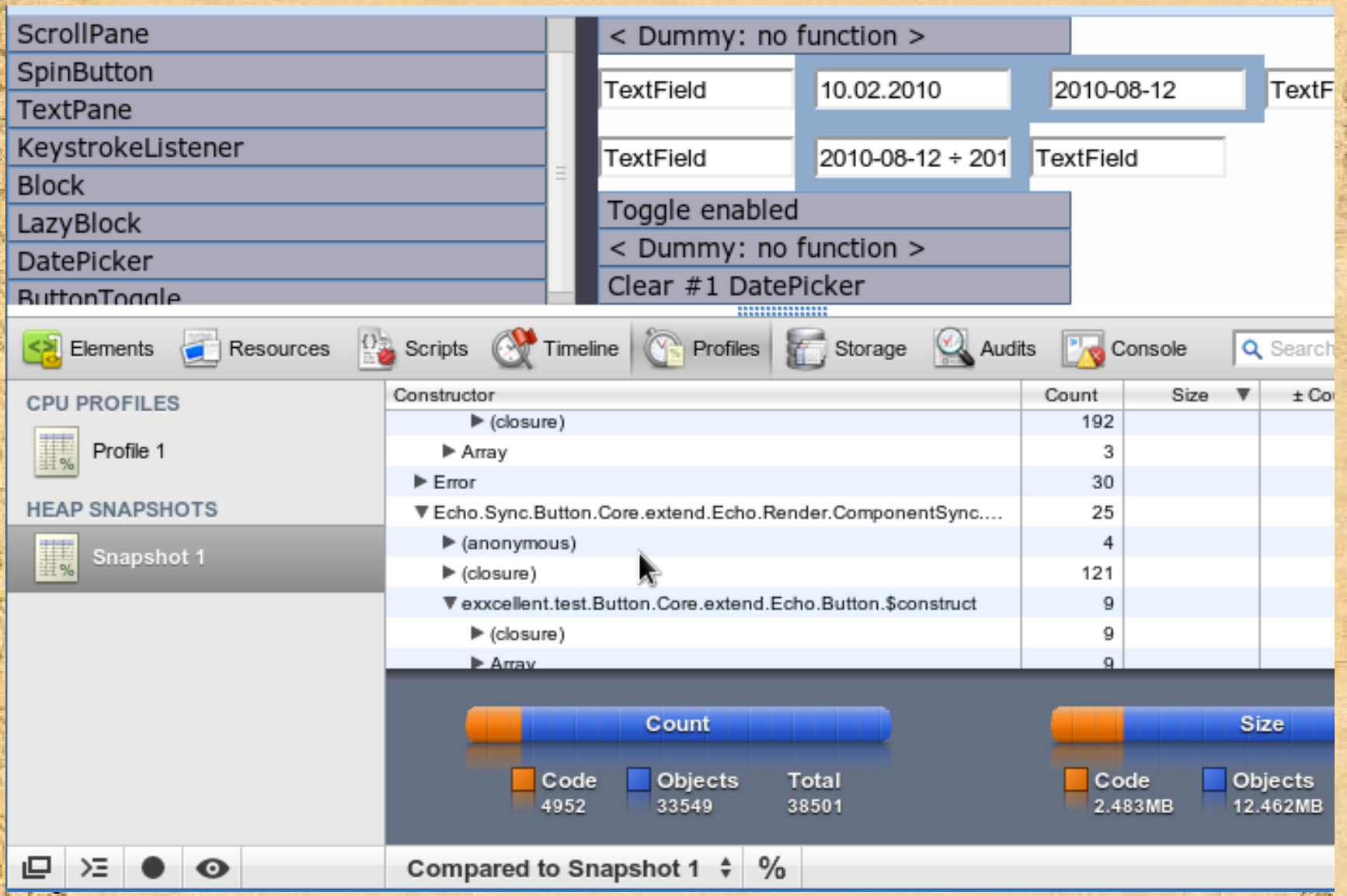
IntelliJ 9 – Refactoring (Renaming)



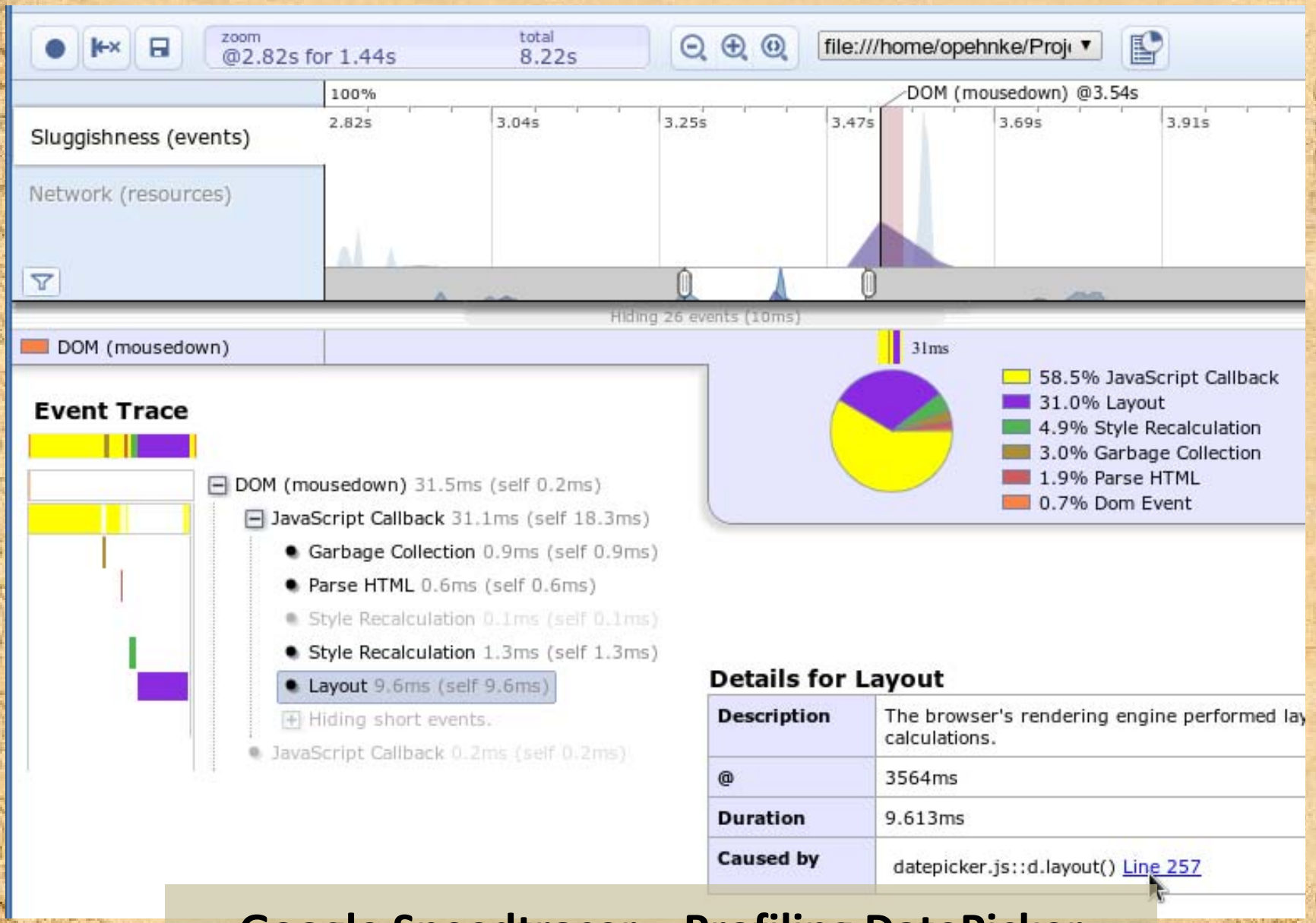
Firefox (Firebug) – HTML Stuktur



Chrome – CPU Profiling



Chrome – Memory Profiler



Google Speedtracer – Profiling DatePicker

Empfehlung des Hauses

Akt II

Aspekt	Lösung
Code Checks	**IntelliJ**, JsLint
IDE	**IntelliJ**, Eclipse + Spket
Strukturierung	**Echo3**
Bibliotheken	**jQuery**, YUI
Profiling	**Firebug**, Google Speed Tracer
Debugging	**IntelliJ**, FireFox FireBug, Webkit Drosera
Testing	**(Selenium**, JsUnit)
IE Voodoo	**...**
Cross Browser	**VMware + IE Collection + Browser**

Grundrezept (gelingt immer)

Akt II

1. **Sprache lernen**
2. **Sprachkonstrukte verstehen!**
3. **Richtige Entwicklungs-plattform**
4. **Vorsicht bei Fertigback-mischungen** (Frameworks)
5. **Sauber strukturieren**

Akt III Die dunklen Seiten

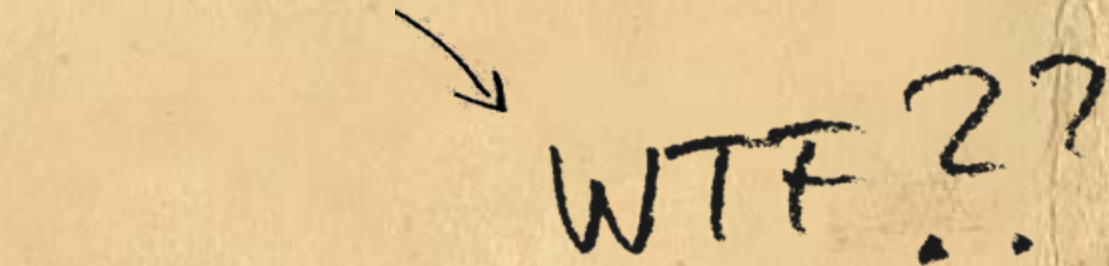
Scope in ES

```
function playScope(param) {
    if (param) {
        var success = true;
    }
    return success;
}
```

Function Scope!

```
function playScope(param) {
    if (param) {
        var success = true;
    }
    return success;
}
```

Akt III



Assoziativgesetz

Akt III

```
a = 0.1;
b = 0.2;
c = 0.3;
```

`(a + b) + c === a + (b + c)`

?

Assoziativgesetz

Akt III

```
a = 0.1;
b = 0.2;
c = 0.3;
(a + b) + c === a + (b + c)
```

{false}

Assoziativgesetz (Lösung)

```
a = 0.1 * 100;
b = 0.2 * 100;
c = 0.3 * 100;
```

(a + b) + c === a + (b + c)

{true}

Bugs that never die

null ist ein object,

Akt III

Warum:

Microsofts JScript Reverse Engineering

Wurde Standard, weil:

„ If we fix that, it could break an program – and at Microsoft we can not tolerate that.“

Wahrheit oder Risiko?

```
var value = „0“;
if (value) {
        magic();
}
```

?

Akt III

Wahrheit oder Risiko?

```
var value = „0“;
if (value) {
        magic();
}
```

{true}

Akt III

Das Ungleichnis

Vergleich	Ergebnis
'' == '0'	False
0 == ''	True
0 == „0“	True
false == „false“	False
false == '0'	True
false == undefined	False
false == null	False
null == undefined	True
„ \t\r\n == 0	True

Akt III

The worst part

... be well prepared

IE Zitat

Don't use closures unless you really need closure semantics.

Akt III

In most cases, non-nested functions are the right way to go.

Eric Lippert, Microsoft

IE Summary

Akt III

Akt III

Und es gibt noch **viel**

mehr zu **entdecken**!

Besuchen Sie uns:

ex|Xcellent solutions

Epilog

Is **Javascript** The Next **Big** Thing?!

JS - The Next Big Thing?

- Web 2.0 lebt & boomt!
- Flash, WebStart, JavaFX, ...
- Neues JS-Universum:
  - Reine JS Komponenten & Frameworks
  - ServerSide JS
  - Run: JS-Performance (IE9: GPU)
  - JS-basierter Flash-Interpreter
- Zunehmende Bedeutung auch im Enterprise-Bereich

Hope:

Java Entwickler sind die ***besseren*** **Javascript** Entwickler.

Fin

Referenzen:

Secrets of Closures, Fronteers 2008, Stuart Langridge

ServerSide Javascript, Steve Yegge

YUI Theater,

Douglas Crockford on Javascript (Google Video)

Buch: „Javascript – The Good Parts“, Douglas Crockford

Zeichnungen:

Oliver Pehnke