



**Das** *lustige*  
*Überlebenshandbuch*  
*für* **JavaScript**

Oliver Pehnke, Benjamin Schmid

**ex|Xcellent**  
solutions

# JavaScript

ist die **erfolgreichste** Sprache

**aller Zeiten**

im bekannten **Universum.**



nasa.org

# Inhaltsverzeichnis

- Prolog

- **Akt I** Sprache

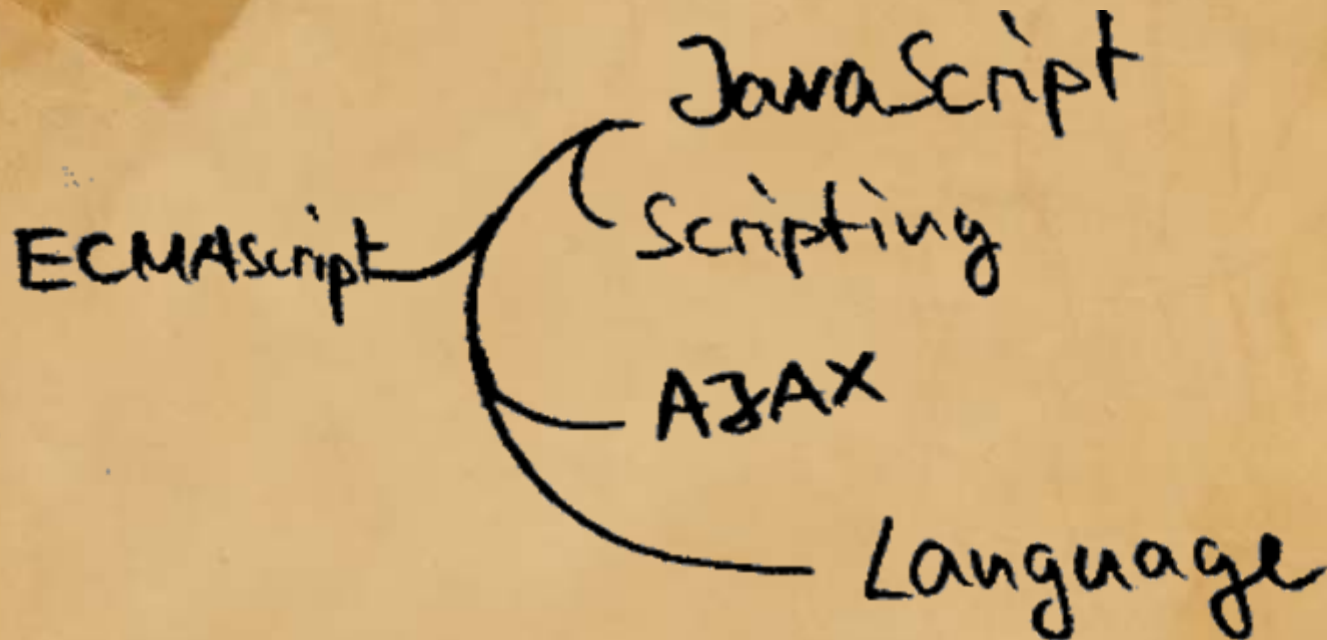
- **Akt II** Werkzeuge

- **Akt III** Die dunklen Seiten

- Epilog

# Prolog

# Evolution von Javascript



# Ein erster Vergleich



- Klassen-basiert
- statisch und stark typisiert

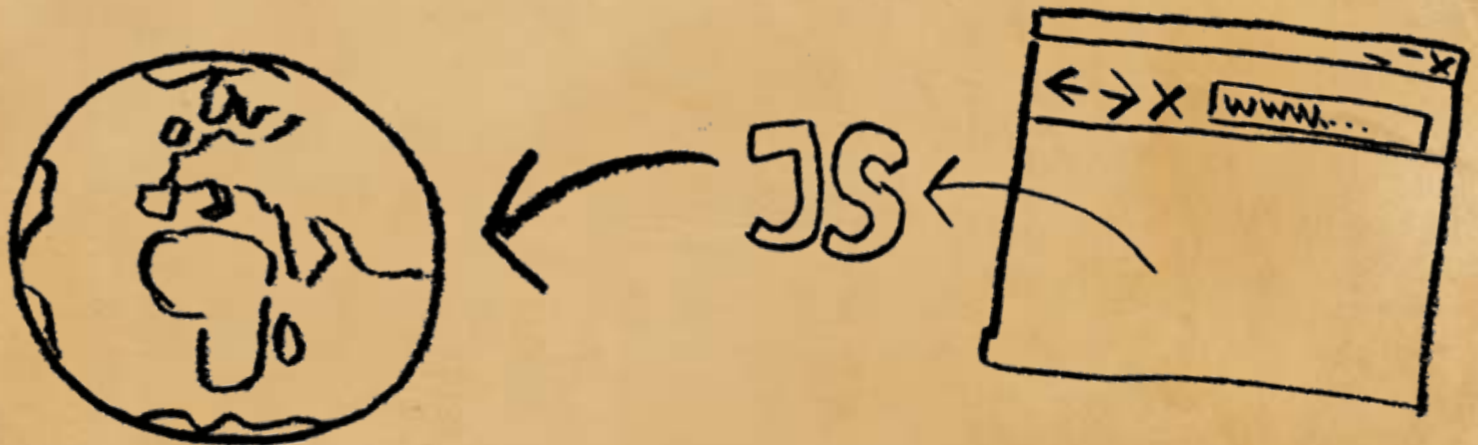


- Prototype-basiert
- dynamisch und schwach typisiert

# 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)

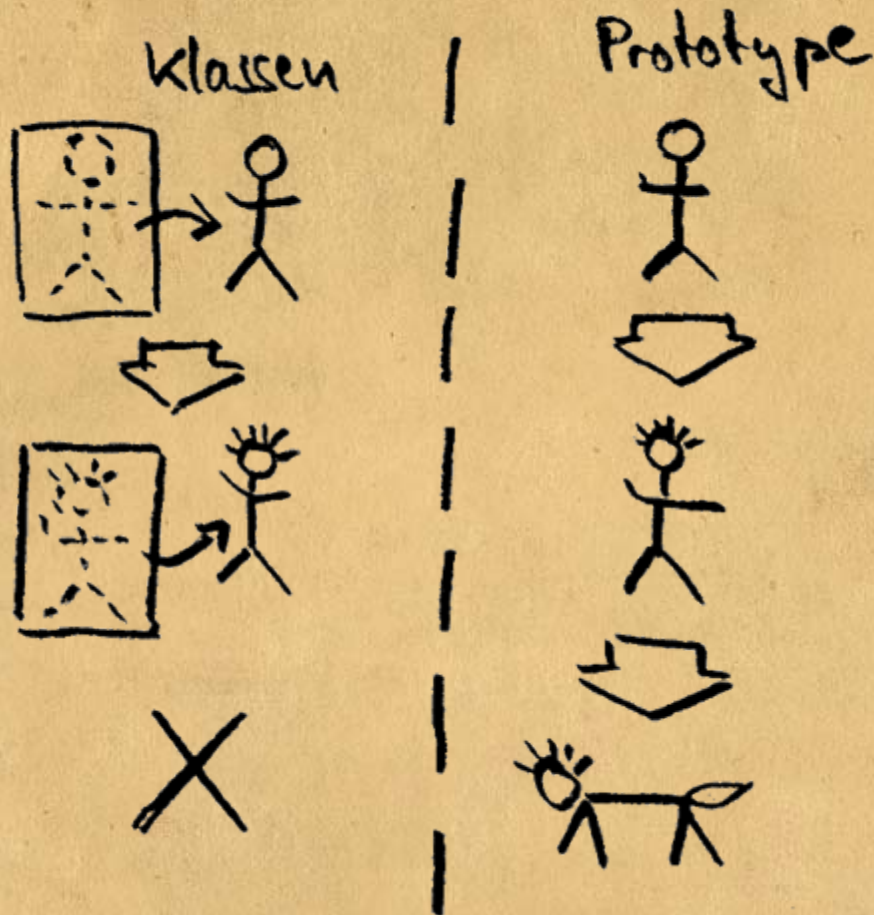




# Akt I Sprache

# 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

Die Schlacht um die  
Vorherrschaft  
hat begonnen!!!



# Datentypen

Akt I

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

# Beispiel First Class Function

Akt I

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

# Closures

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



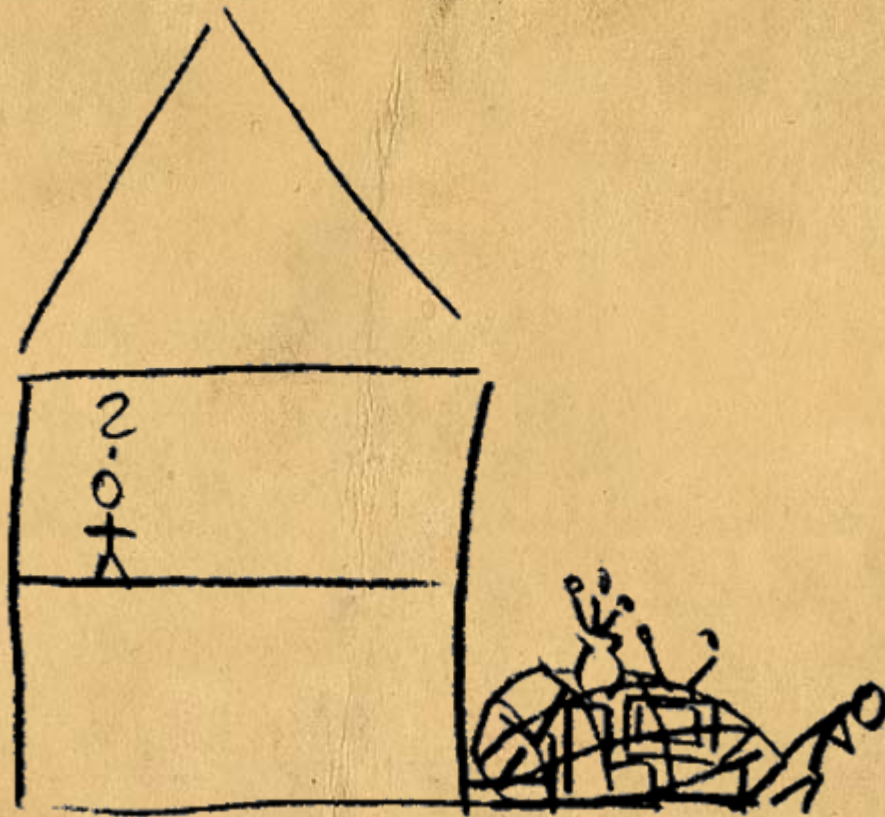
# Closures

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

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

| Aufruf             | this                        |
|--------------------|-----------------------------|
| <i>function</i>    | global object<br>undefined  |
| <i>method</i>      | object (Eigner der Methode) |
| <i>constructor</i> | Das „new“ object            |
| <i>apply</i>       | argument                    |

# Prototypische OO & Closures

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

***{Hello Struppi.}***

# Und was ist mit OO?

```
function Container(param) {
```

```
  function dec() {  
    if (secret > 0) {  
      secret -= 1;  
      return true;  
    } else { return false; }  
  }
```

```
  var secret = 3;
```

private

```
  var that = this;
```

```
  this.service = function () {  
    if (dec()) {  
      return that.secret;  
    } else { return null; }  
  }  
};
```

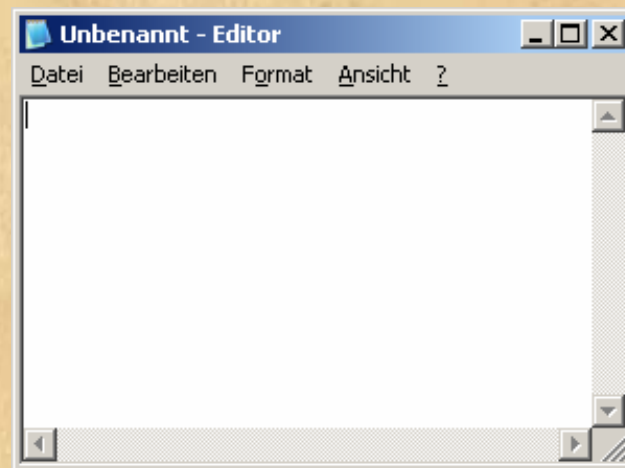
privileged

```
}
```

# **Akt II** **Werkzeuge**

# Werkzeuge: Was ist zu erwarten?

Akt II



# Werkzeuge: Was ist zu erwarten?

Akt II

Code Completion

Logging

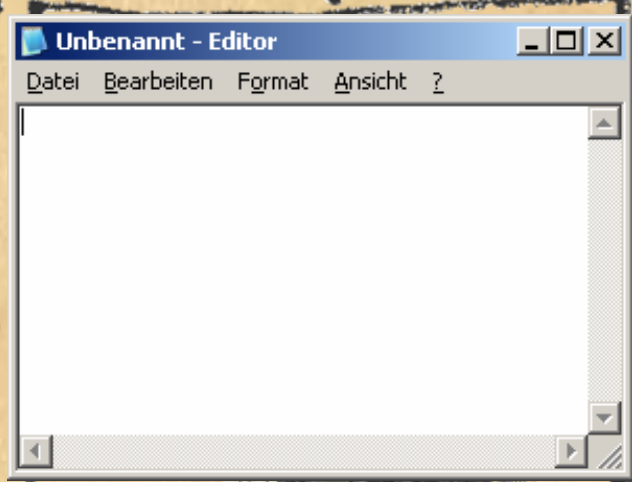
Testing

Static Code Checks

Profiling

Deployment

Debugging



# Die Küche empfiehlt:

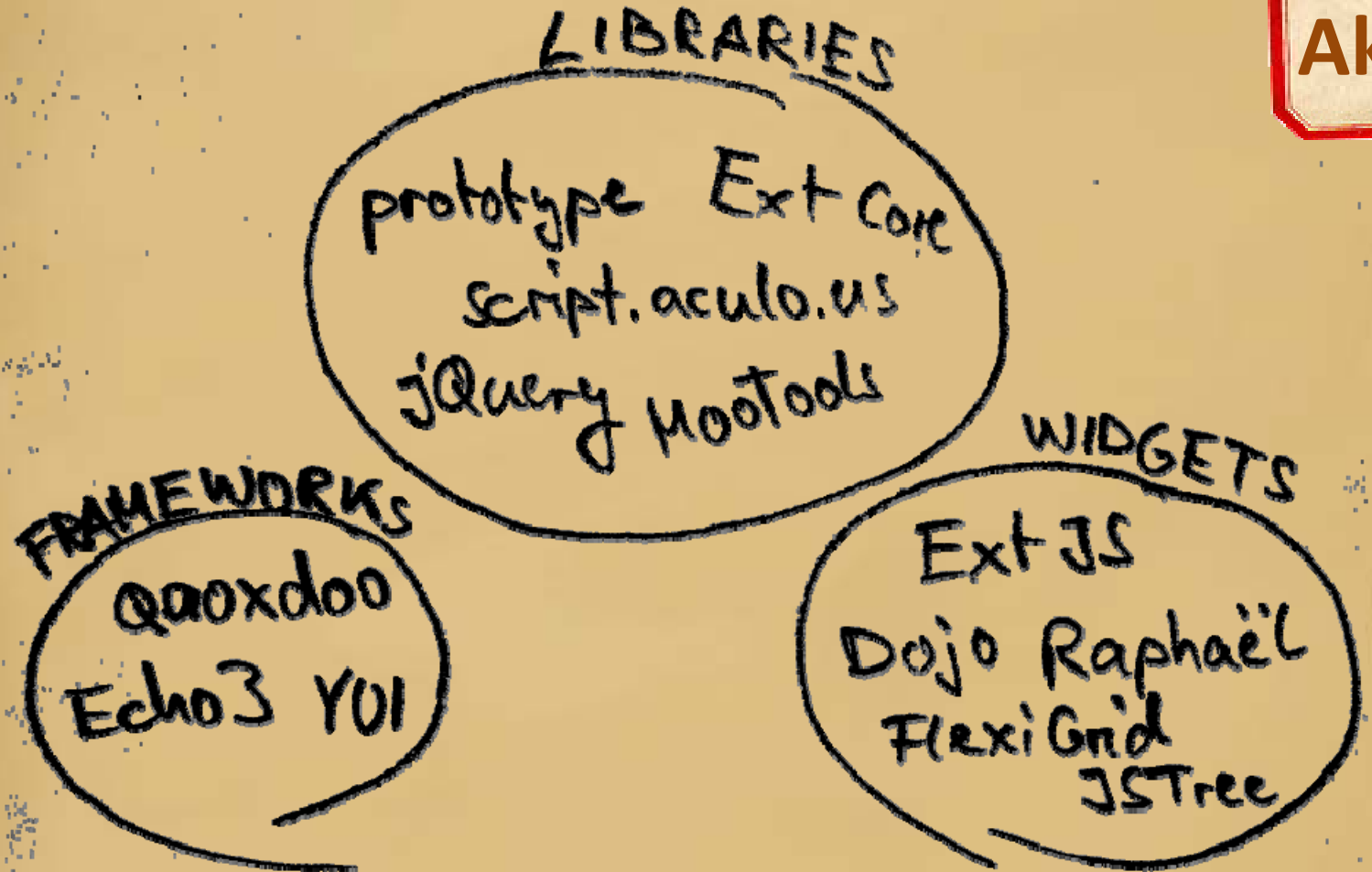
Akt II

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



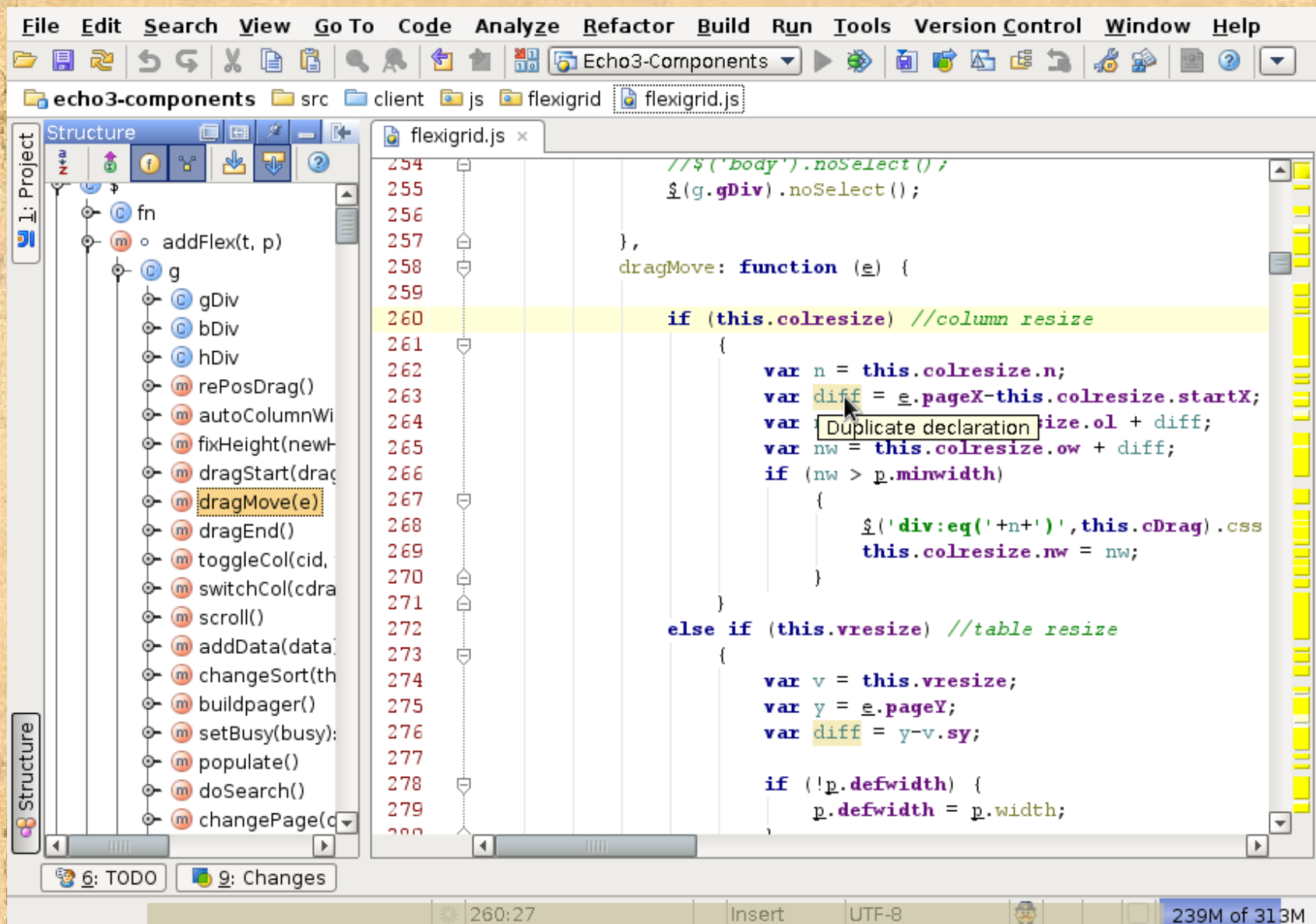
# Die Wahrheit über Libraries

Akt II

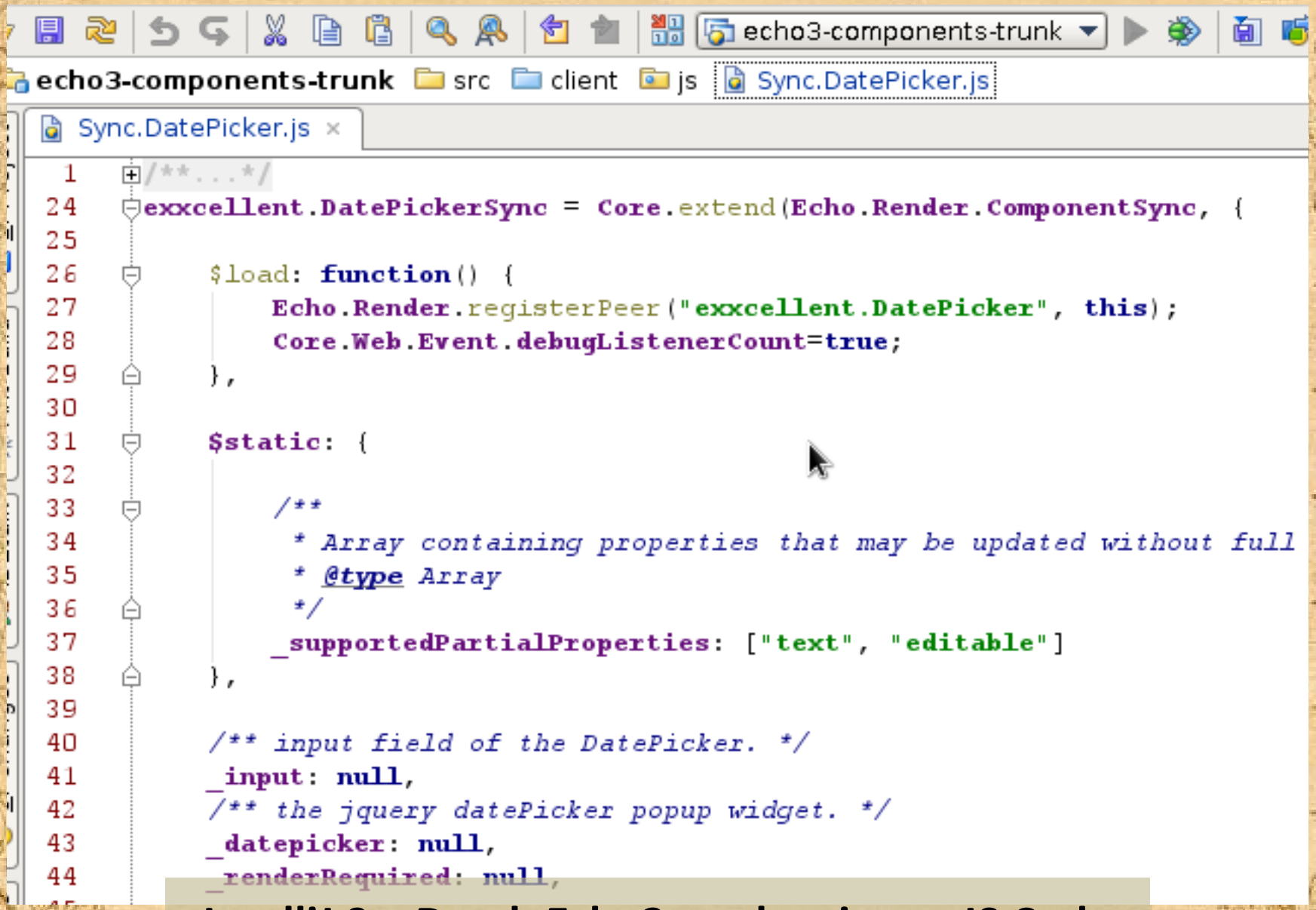


Akt II

*Demo*



## IntelliJ 9 – Codestruktur, Inspections



```
1  /**...*/
24  excellent.DatePickerSync = Core.extend(Echo.Render.ComponentSync, {
25
26      $load: function() {
27          Echo.Render.registerPeer("excellent.DatePicker", this);
28          Core.Web.Event.debugListenerCount=true;
29      },
30
31      $static: {
32
33          /**
34           * Array containing properties that may be updated without full
35           * @type Array
36           */
37          _supportedPartialProperties: ["text", "editable"]
38      },
39
40      /** input field of the DatePicker. */
41      _input: null,
42      /** the jquery datePicker popup widget. */
43      _datepicker: null,
44      _renderRequired: null,
45  }
```

## IntelliJ 9 – Durch Echo3 strukturierter JS Code

File Edit Search View Go To Code Analyze Refactor Build Run Tools Version Control

echo3-components-trunk src client js Sync.DatePicker.js

Sync.DatePicker.js x

```
65  /**
66   * Describes how a component is initially built.
67   */
68   renderAdd: function(update, parentElement) {
69       if (window.console && window.console.log) { window.console.log('
70       /* the input field for the date. */
71       this._input = document.createElement("input");
72       this._input.id = this.component.renderId;
73       if (!this.component.render(excellent.DatePicker.EDITABLE, true)
74       this._input.readOnly = true;
75   }
76   this._input
77   var maximum
78   if (maximum
79       this._
80   }
81   this._rend
82   }
83   // add eve
84   Core.Web.E
```

|                      |         |
|----------------------|---------|
| constructor          | Object  |
| hasOwnProperty       | Boolean |
| isPrototypeOf        | Boolean |
| length               | Number  |
| propertyIsEnumerable | Boolean |
| toLocaleString       | Object  |
| toSource             | Object  |
| toString             | String  |
| unwatch              | void    |
| valueOf              | String  |

, this.

## IntelliJ 9 – Best Guess Code Completion

file:///home/opehn...t/test/index.html

|                   |                                 |
|-------------------|---------------------------------|
| FlexiGrid         | Empty                           |
| ScrollPane        | TextField 2010-02-12 10.01.2010 |
| SpinButton        |                                 |
| TextPane          | TextField TextField             |
| KeystrokeListener | Toggle enabled                  |
| Block             | Empty                           |
| LazyBlock         | Clear #1 DatePicker             |
| DatePicker        |                                 |
| ButtonToggle      |                                 |

Console HTML CSS Script DOM Net

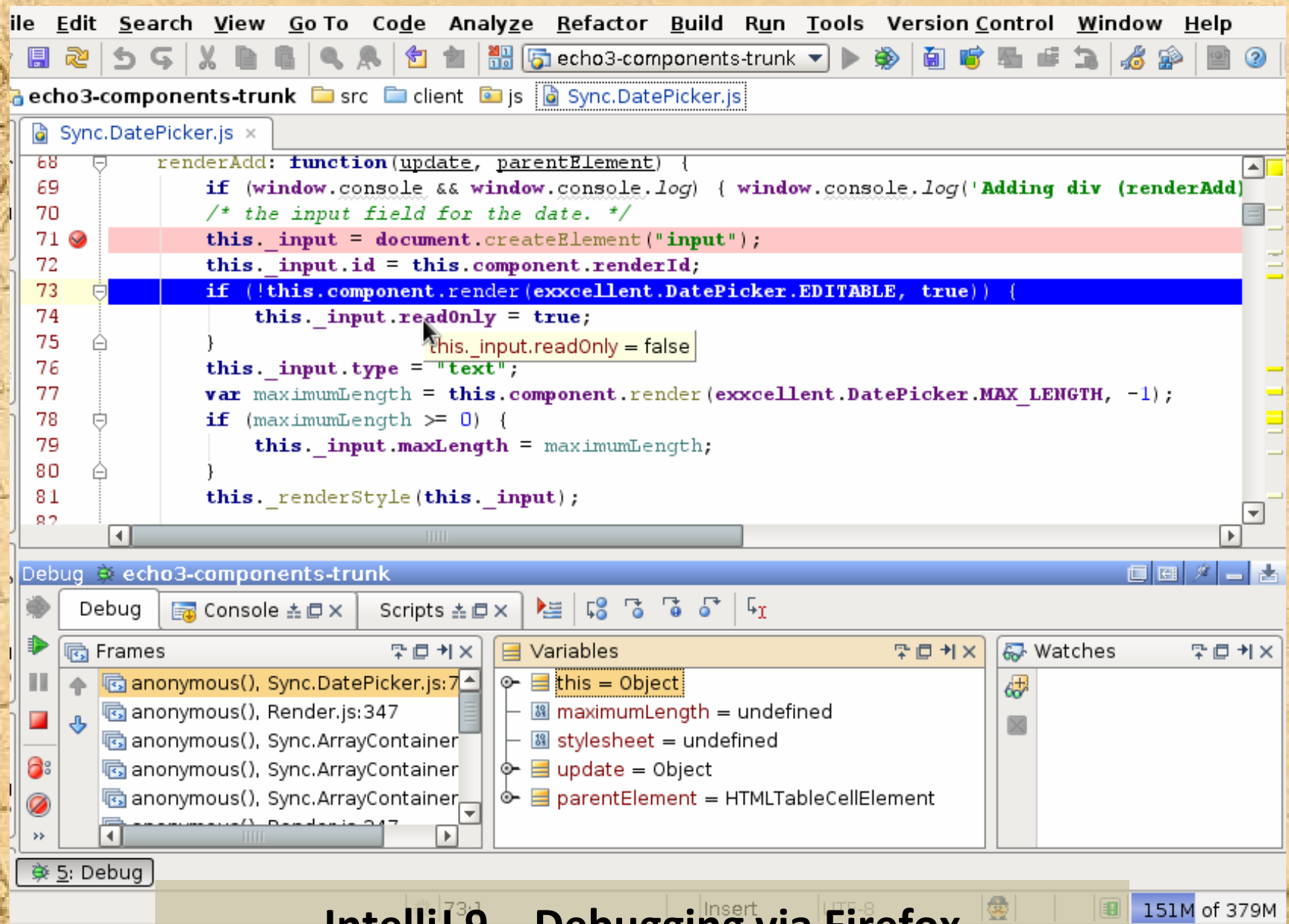
Clear Persist Profile

```

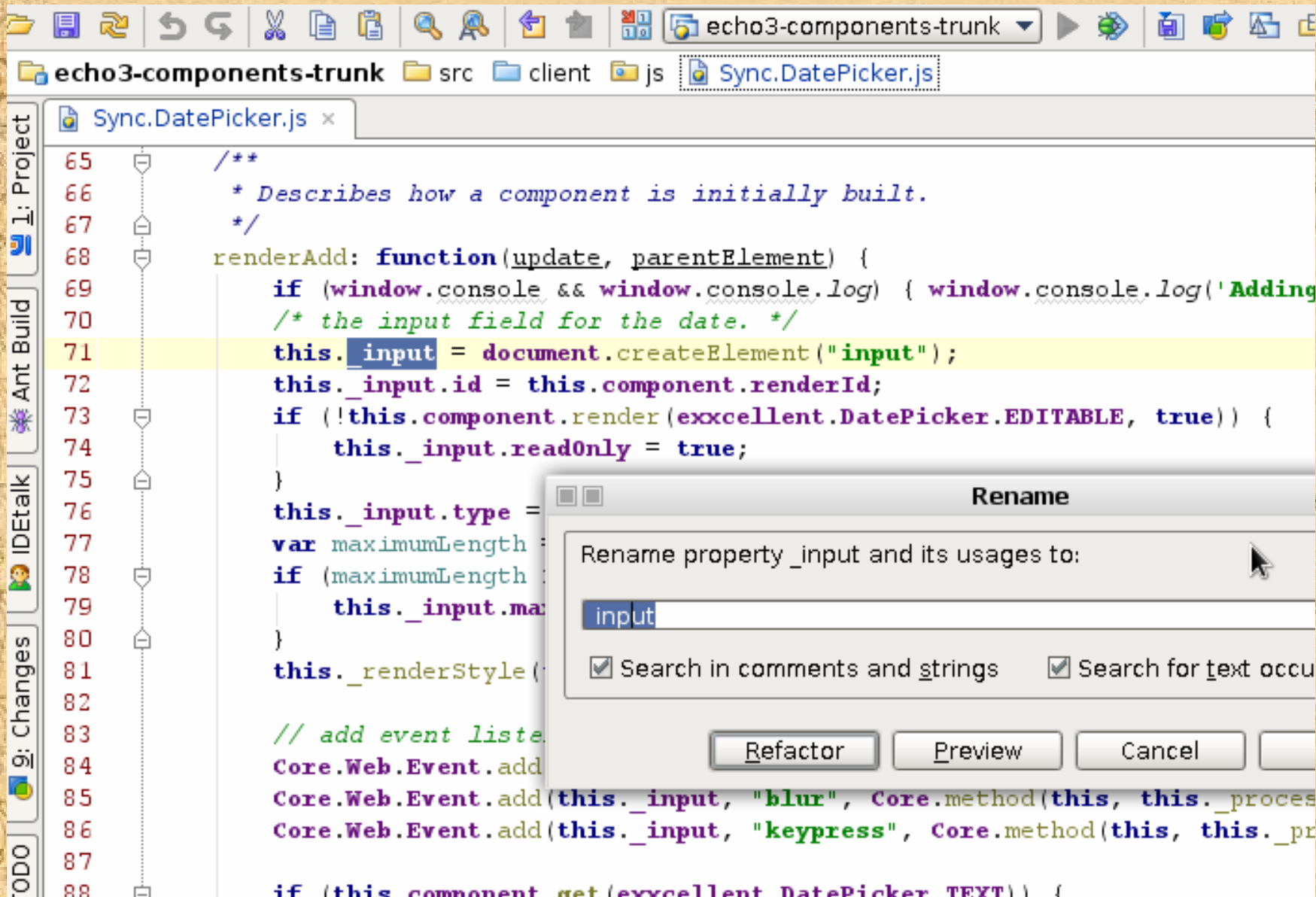
Displaying DatePicker (renderDisplay()).
Creating new DatePicker (renderDisplay()).
Adding div (renderAdd).
Adding div (renderAdd).
Displaying DatePicker (renderDisplay()).
Creating new DatePicker (renderDisplay()).
DatePicker.init() mode: single
DatePicker.init(), new id: datepicker_639

```

## Firefox (Firebug) - Console



## IntelliJ 9 – Debugging via Firefox



## IntelliJ 9 – Refactoring (Renaming)

file:///home/oepn...t/test/index.html

FlexiGrid  
ScrollPane  
SpinButton  
TextPane  
KeystrokeListener  
Block  
LazyBlock  
DatePicker  
ButtonToggle

Empty  
TextField 2010-02-12 10.01.2010 TextField  
TextField TextField  
Toggle enabled  
Empty  
Clear #1 DatePicker

Console HTML CSS Script DOM Net

Edit input#ex...ePickerA < td < tr < tbody < table < div#CL.7 < div

```

0px;">
<td style="background-
color: rgb(142, 171,
204); padding: 5px
10px;">
  <input id="xxxcellentUnitTestDataPickerA"
  align: left; width:
  100px;">
</td>
<td style="background-
color: rgb(142, 171,
204); padding: 5px
10px;">
<td style="padding:

```

element.style {  
text-align: left;  
width: 100px;  
}  
Inherited from table  
element.style {  
border-collapse: collapse;  
}  
Inherited from body  
element.style {  
font-family: verdana,arial,helveti  
serif;  
font-size: 10pt;  
}

## Firefox (Firebug) – HTML Struktur

Chrome DevTools interface showing CPU Profiling results. The top panel displays the current state of the application, including a date picker and a calendar. The bottom panel shows the CPU Profiling table, which lists the functions being executed and their associated CPU time.

| Self   | Total   | Function                                                                                      |
|--------|---------|-----------------------------------------------------------------------------------------------|
| 58.33% | 100.00% | (program)                                                                                     |
| 8.33%  | 8.33%   | ▶ _processComponentPropertyUpdate <a href="#">Application.js:2465</a>                         |
| 8.33%  | 8.33%   | ▶ data <a href="#">jquery-1.3.2.js:1274</a>                                                   |
| 8.33%  | 16.67%  | ▶ unique <a href="#">jquery-1.3.2.js:1113</a>                                                 |
| 8.33%  | 8.33%   | ▼ d.tmpl <a href="#">datepicker.js:584</a>                                                    |
| 8.33%  | 8.33%   | ▼ d.fill <a href="#">datepicker.js:66</a>                                                     |
| 8.33%  | 8.33%   | ▼ d.showCalendar <a href="#">datepicker.js:447</a>                                            |
| 8.33%  | 8.33%   | ▼ inputFieldClick <a href="#">datepicker.js:899</a>                                           |
| 8.33%  | 8.33%   | ▼ jQuery.event.handle <a href="#">jquery-1.3.2.js:2665</a>                                    |
| 8.33%  | 8.33%   | ▼ handle.jQuery.data.elem.handle.jQuery.data.elem.handle <a href="#">jquery-1.3.2.js:2464</a> |
| 8.33%  | 8.33%   | (program)                                                                                     |
| 8.33%  | 8.33%   | ▶ topStackFrame                                                                               |
| 0%     | 8.33%   | ▶ inputFieldClick <a href="#">datepicker.js:899</a>                                           |
| 0%     | 25.00%  | ▶ calendarSelect <a href="#">datepicker.js:889</a>                                            |

Heavy (Bottom Up) %

## Chrome – CPU Profiling

ScrollPane

SpinButton

TextPane

KeyListener

Block

LazyBlock

DatePicker

ButtonToggle

< Dummy: no function >

TextField

10.02.2010

2010-08-12

TextF

TextField

2010-08-12 ÷ 201

TextField

Toggle enabled

< Dummy: no function >

Clear #1 DatePicker

Elements

Resources

Scripts

Timeline

Profiles

Storage

Audits

Console

Search

CPU PROFILES

Profile 1

HEAP SNAPSHOTS

Snapshot 1

| Constructor                                                  | Count | Size | ± Co |
|--------------------------------------------------------------|-------|------|------|
| ▶ (closure)                                                  | 192   |      |      |
| ▶ Array                                                      | 3     |      |      |
| ▶ Error                                                      | 30    |      |      |
| ▼ Echo.Sync.Button.Core.extend.Echo.Render.ComponentSync.... | 25    |      |      |
| ▶ (anonymous)                                                | 4     |      |      |
| ▶ (closure)                                                  | 121   |      |      |
| ▼ excellent.test.Button.Core.extend.Echo.Button.\$construct  | 9     |      |      |
| ▶ (closure)                                                  | 9     |      |      |
| ▶ Array                                                      | 9     |      |      |

Count

Size

Code

Objects

Total

4952

33549

38501

Code

Objects

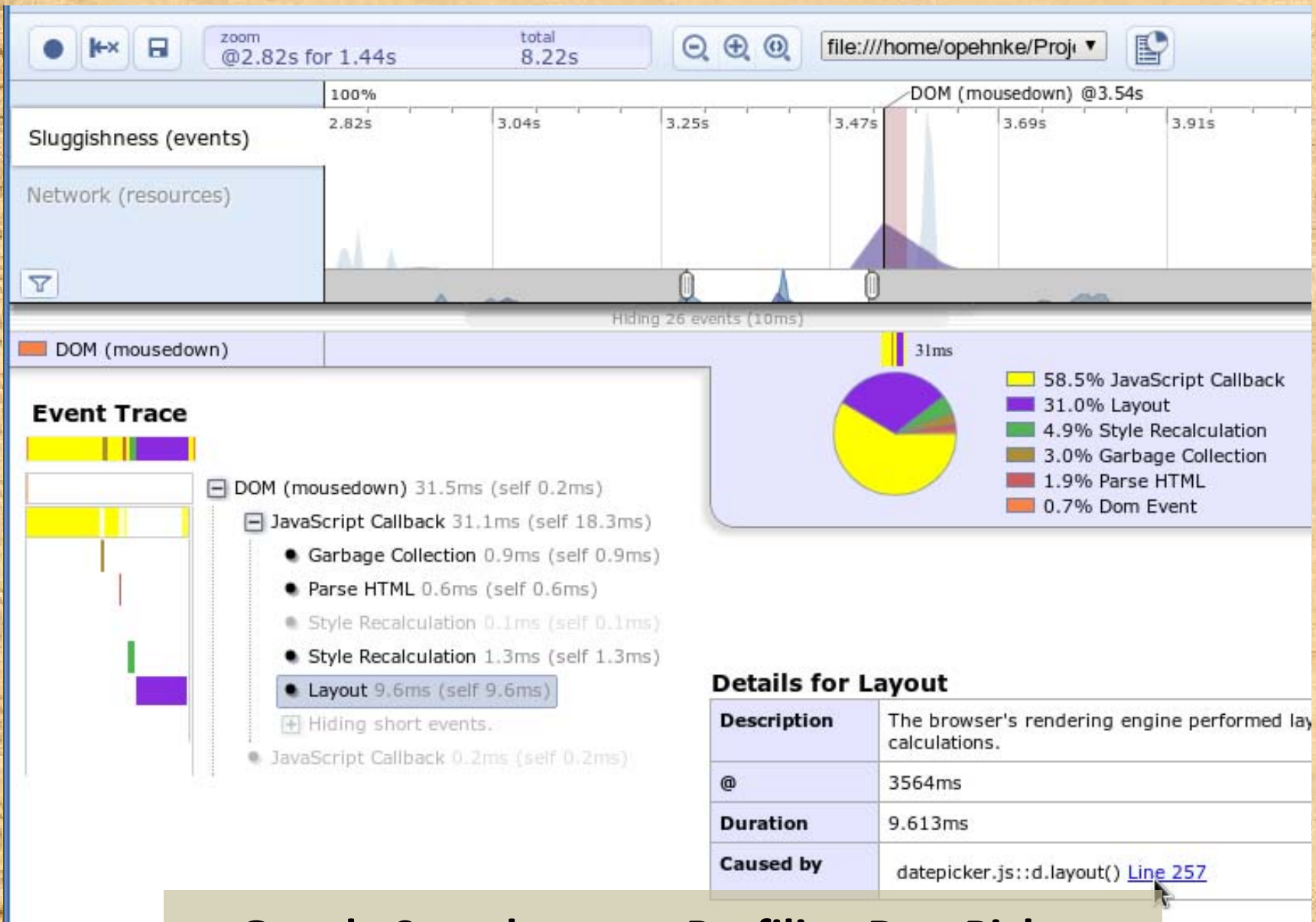
2.483MB

12.462MB

Compared to Snapshot 1

%

## Chrome – Memory Profiler



## Google Speedtracer – Profiling DatePicker

# Empfehlung des Hauses

**Akt II**

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

# 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;  
}
```

**Akt III**

# Function Scope!

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

Akt III

WTF??

# Assoziativgesetz

$$a = 0.1;$$

$$b = 0.2;$$

$$c = 0.3;$$

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

?

Akt III

# Assoziativgesetz

$a = 0.1;$

$b = 0.2;$

$c = 0.3;$

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

***{false}***

Akt III

# Assoziativgesetz (Lösung)

$a = 0.1 * 100;$

$b = 0.2 * 100;$

$c = 0.3 * 100;$

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

***{true}***

Akt III

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 |
|---------------------------------|----------|
| <code>" == '0'</code>           | False    |
| <code>0 == "</code>             | True     |
| <code>0 == „0“</code>           | True     |
| <code>false == „false“</code>   | False    |
| <code>false == '0'</code>       | True     |
| <code>false == undefined</code> | False    |
| <code>false == null</code>      | False    |
| <code>null == undefined</code>  | True     |
| <code>„ \t\r\n == 0</code>      | 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