AJAX IN ACTION

JavaScript

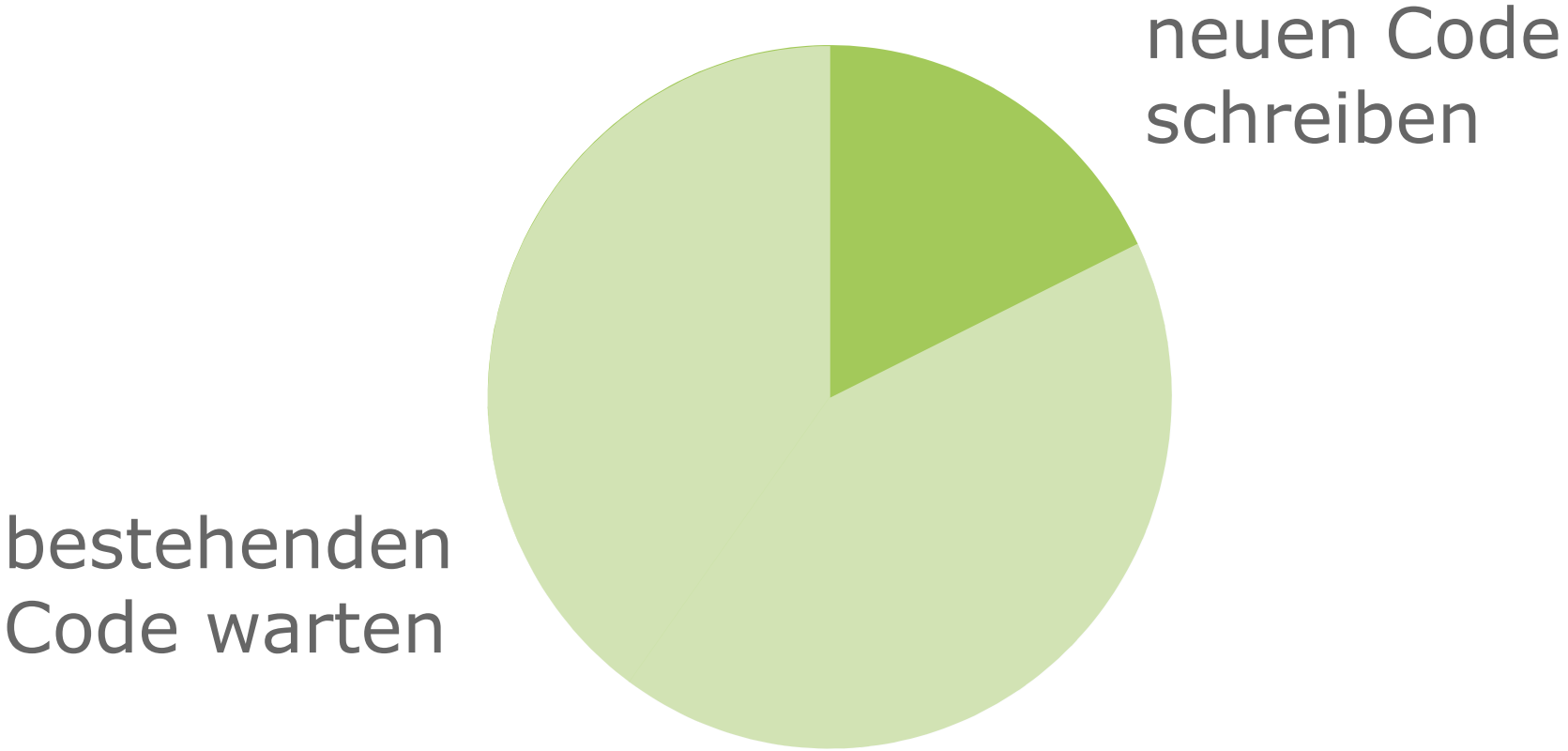
dokumentieren
testen
validieren

Richard Sternagel
1&1 Internet AG
Mainz, 30.10.08

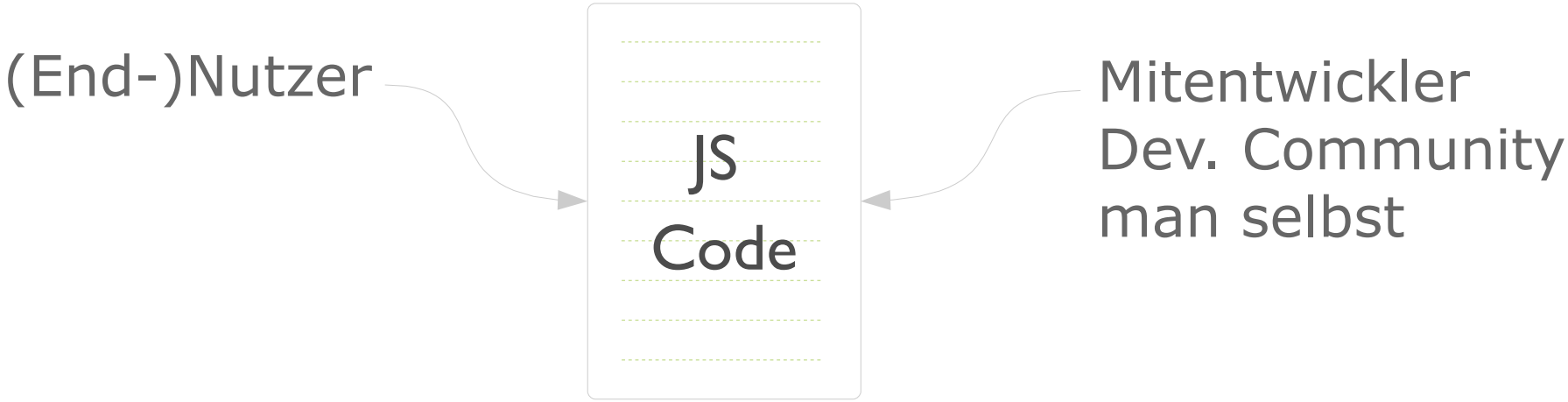
Agenda

- Motivation
- Frameworks/Tools zum …
  - dokumentieren
  - testen
  - validieren
- Fazit

Wozu der ganze Aufwand?



Wozu der ganze Aufwand?



Optimierung:

- zusammenfassen
- minimieren
- komprimieren

Optimierung:

- Wartbarkeit!

Was ist wartbarer Code?

- verständlich
- anpassbar
- erweiterbar
- testbar
- modular

- dokumentieren
- testen
- validieren
- …

Beipackzettel

- nicht jede Technik bei jedem Projekt
- nicht alles gleichzeitig einführen
- was funktioniert für mich/das Team
- am Anfang Mehraufwand
  - ... der sich aber auszahlt!

Agenda

- ~~Motivation~~
- Frameworks/Tools zum …
  - dokumentieren
  - testen
  - validieren
- Fazit

Dokumentations Tools

- JSDoc
- JsDoc Toolkit
- ScriptDoc

Exkurs: Doc Comments in Java

```
/**
  * Logs a user with the given userName in.
  *
  * @param  userName  The name of the user
  * @author  John Doe
  * @see       The <a href="...">Link</a>
  */
public void logIn(String userName) {
    ...
}
```

tags

doc comment

Tipp: How to Write Doc Comments for the Javadoc Tool

JsDoc Toolkit (Michael Mathews)

- in JavaScript geschrieben
- kein fixes Output-Format (out of the box = HTML)
- wird mittels Rhino ausgeführt
- Templates
- Plugins

JavaScript Implementierung in Java

JsDoc Tk: Syntax Beispiele

- ```
  /**
   *  @param  {String} Absolute/Relative Path to a file.      // {String|Number}
   *  @returns {Array}  Lines from the file.                  // {String[]}
   */                                                         // [String]
  function readLines(filepath) {}                             // [String="/home"]
  ```

- ```
  function readLines(/**String*/ filepath) /**Array*/ {}
  ```

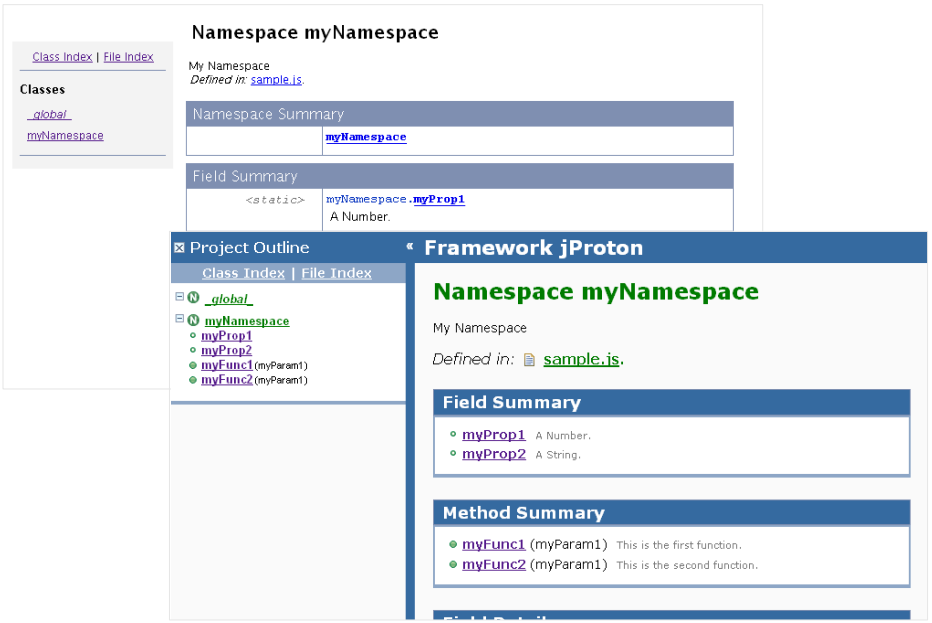
- ```
  /**
   * @param userInfo Information about the user.
   * @param userInfo.name The name of the user.
   * @param userInfo.email The email of the user.
   */
  function logIn(userInfo) {)
  ```
  ```

JsDoc Tk: Tag Referenz

- @author
- @augments|@extends
- @borrows that as this
- @class
- @constant
- @constructor
- @constructs
- @default
- @deprecated
- @description
- @example
- @event
- @field
- @fileOverview
- @function
- @inner
- @ignore
- @lends|@scope
- @memberOf
- @name
- @namespace
- @param
- @private
- @property
- @public
- @requires
- @return|returns
- @see|{@link ...}
- @since
- @static
- @throws|@exception
- @type
- @version

JsDoc Tk erweitern

- Templates
  - HTML (JSON, XML)
  - publish.js
    - publish(symbolSet) {}
  - JsPlate
- Plugins
  - Ausführung vor Templategenerierung
  - Eventbasiert (onSymbol, onDocCommentTags, …)



JsDoc Tk: Encoding

- in allen JS-Dateien gleiches Encoding!
- UTF-8 für doc files (default)
  <meta http-equiv="content-type" content="text/html; charset=utf-8">

  Cmdline Option: "-e=iso-8859-1" <...charset=iso-8859-1">

JsDoc Tk: Kommandozeile

```
$ cd path/to/jsdoctk
$ java -jar jsrun.jar app/run.js [OPTIONS] <SRC_DIR> ...

{jsdoctk} => java -jar jsrun.jar app/run.js -t=templates/jsdoc

{jsdoctk} myscripts/ single/file.js
{jsdoctk} -h
{jsdoctk} -r=4 myscripts/
{jsdoctk} -x=sc,js,txt myscripts/
{jsdoctk} -d=my_docs myscripts/
{jsdoctk} -p myscripts/
{jsdoctk} -a myscripts/
{jsdoctk} -t=templates/myOwnTmpl myscripts/
{jsdoctk} -c=sample.conf myscripts/
```

Agenda

- ~~Motivation~~
- ~~Frameworks/Tools zum ...~~
  - ~~dokumentieren~~
  - testen
  - validieren
- Fazit

(Unit) Testing Frameworks

- JsUnit (Hieatt)
- JsUnit (Schaible)

- YUI Test
- QUnit
- DOH
- Scriptaculous Unit Testing

- Test.Simple
- Test.More

- J3Unit
- JSMock
- Mock4JS
- JSSpec
- Crosscheck

Unit Test (Modul-/Komponententest)

- verifiziert Verhalten einer Unit
- Unit bei OOP = Klasse / Methode
- idealerweise isoliert (unabhängig von anderen Obj.)

```
Erwartet  ?=  Verhalten
    5     ?=  add(2, 3)
```

Ziel: Automatisierung

YUI Test (Nicholas C. Zakas)

- einfache Syntax / gut dokumentiert
- separat von YUI einsetzbar
- Event Simulation
- Asynchrones Testen
- Reporting » Automatisierung
- Eventbasiert (einklinken möglich)
- eigene Visualisierung möglich

YUI Test: Überblick

- TestCases
- TestSuite
- setUp()
- tearDown()
- _should.ignore
- _should.error
- fail()

- areEqual() ==
- areNotEqual() !=
- areSame() ===
- areNotSame() !==

- isArray()
- isBoolean()
- isFunction()
- isNumber()
- isObject()
- isString()

- isTypeOf()
- isInstanceOf()

- isFalse()
- isTrue()
- isNaN()
- isNotNaN()
- isNull()
- isNotNull()
- isUndefined()
- isNotUndefined()

YUI Test: Events simulieren I

- YAHOO.util.UserAction
  - .click()
  - .dblclick()
  - .mousedown()
  - .mouseup()
  - .mouseover()
  - .mouseout()
  - .mousemove()
  - .keyup()
  - .keydown()
  - .keypress()

YUI Test: Events simulieren II

- YAHOO.util.UserAction.click(myElem);
- YAHOO.util.UserAction.click("elemId");

- YAHOO.util.UserAction.click("elemId", {
  clientX: 10,
  clientY: 20
  });

- YAHOO.util.UserAction.keypress("elemId", {
  charCode: 13
  });

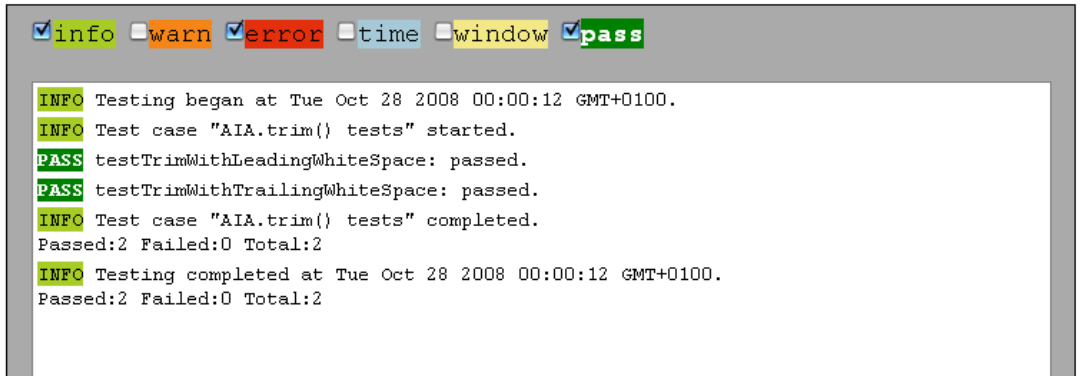
YUI Test: TestRunner Events

- pass
- fail
- ignore

- begin
- complete

- testcasebegin
- testcasecomplete

- testsuitebegin
- testsuitecomplete



YUI Test: Asynchrones Testen

- wait(func, delay);
- wait(timeout) und später resume(func);

- Keine wirklichen Ajax-Responses testen
  - Test: Input an Server korrekt
  - Test: Output vom Server
    - JS-Handling korrekt für alle Responsevarianten?

YUI Test: Testreporter

```
YAHOO.tool.TestRunner.subscribe("complete", function(event) {
    var reporter = new YAHOO.tool.TestReporter(
                                        "http://www.example.com/report.php"
                                        // , YAHOO.tool.TestFormat.JSON
    );
    reporter.report(event.results);
});
```

YUI Test 3.0

- Mock Objekte
- mehr simulierte Events
- bessere Fehlererkennung
- neue Visualisierungsmöglichkeiten

Agenda

- ~~Motivation~~
- ~~Frameworks/Tools zum ...~~
  - ~~dokumentieren~~
  - ~~testen~~
  - validieren
- Fazit

Code Validatoren / Checker

- JSLint
- … ?

JSLint (Douglas Crockford)

- Tool zur Code Analyse
  - syntax checks
  - code conventions
- striktere Teilmenge von JavaScript

JSLint: Checks

- semicolon / comma
- constructors und new
- for in statement
- with statement
- == / != vs. === / !==
- unreachable code
- ...

JSLint: Einsatzmöglichkeiten

- Formular auf Website remote / lokal
- via Rhino
  - Kommandozeile
  - Eclipse (External tools)
- YSlow (Firefox Extension)
- Apatana Studio
- TextMate Bundle

JSLint steuern

- /*global
  - globale Variablen angeben
- /*member
  - Schreibfehler & fehlende Deklaration identifizieren
  - nur diese erlauben
- /*jslint
  - >20 Optionen, die standardmäßig false sind

JSLint kann ebenso prüfen …

- HTML
  - lowercase & end tags, Verschachtelung, Entities, keine „inline event handler“
  - kein Support für XHTML (XML Deklaration / xml:lang)
  - empfiehlt Verzicht von type="text/javascript" :/
- JSON
  - wenn erstes Zeichen „{“ oder „[“

Agenda

- ~~Motivation~~
- ~~Frameworks/Tools zum ...~~
  - ~~dokumentieren~~
  - ~~testen~~
  - ~~validieren~~
- Fazit

Tipp: Coding Standards / Conventions

- Douglas Crockford / JSLint
- Mozilla Developer Center
- Dojo Styleguide

- Commit Hook » Auto-Konvertierung

Ziel: JS Build Prozess

- validieren
- Coding Standards prüfen
- Tests laufen lassen
- API Dokumentation erzeugen
- minimieren
- ein einzelne Datei erzeugen
- ...

Fazit

- dokumentieren
  - http://jsdoctoolkit.org/
- testen
  - http://developer.yahoo.com/yui/yuitest/
- validieren
  - http://www.jslint.com/

Videos

- Nicholas C. Zakas
  - „Test-Driven Development with YUI Test“
  - „Maintainable JavaScript“
- Christian Heilmann
  - „Scripting Maintainability“

Grafiken

- „Rhino 2“ – Steve Cornish
  - http://flickr.com/photos/scornish/1763558937/
- „Package Settings“ – Everaldo Coelho
  - http://www.everaldo.com/