

Robuste, lesbare Web-UI-Tests mit Page Object Pattern, Geb und Spock

Stephan Classen



Über mich

Stephan Classen

- » TDD enthusiast
- » Liebt Open Source
- » Hasst repetitive Aufgaben



Über mich

Stephan Classen

- » TDD enthusiast
- » Liebt Open Source
- » Hasst repetitive Aufgaben
- » Ein bisschen paranoid



Über mich

stephan.classen@canoo.com

 github.com/sklassen/sessions

Aktuelle Folien



[delivering end-user happiness]

canoo

Agenda

- » Page Object Pattern
- » GEB
- » Spock
- » Tipps & Tricks

Vorteile von UI Tests

- » End To End – Kein Mocking
- » Erkennt Probleme einzelner Browser
- » Schnellere Iterationen
- » Spart Geld

Herausforderungen bei UI Tests

- » Schwierig zu automatisieren
- » Stabilität
- » Anfällig für Änderungen
- » Asynchronität
- » Langsam

Herausforderungen bei UI Tests

- » Schwierig zu automatisieren
- » Stabilität
- » Anfällig für Änderungen
- » Asynchronität
- » Langsam

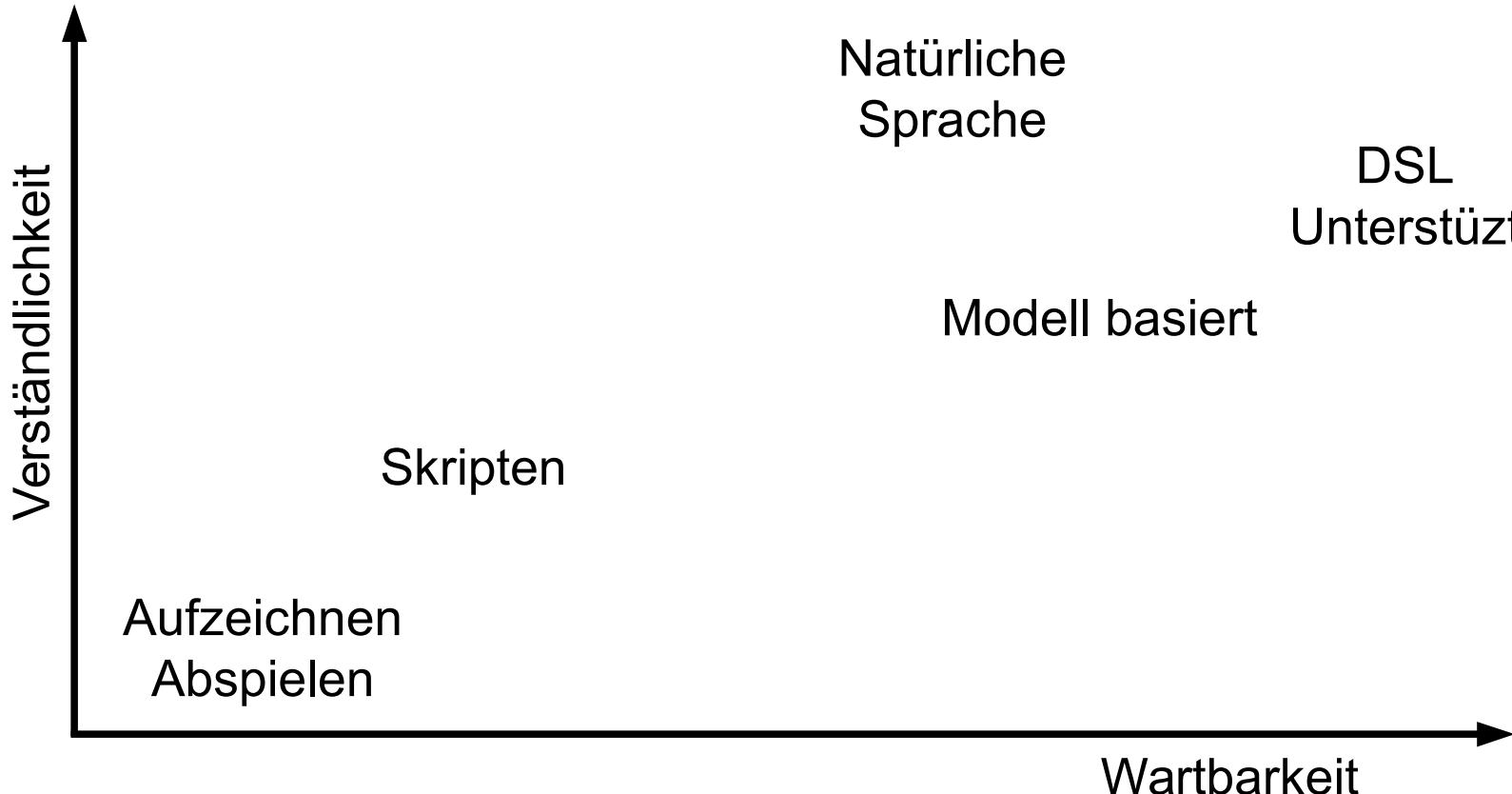
Parallel

Warten,
warten,
warten

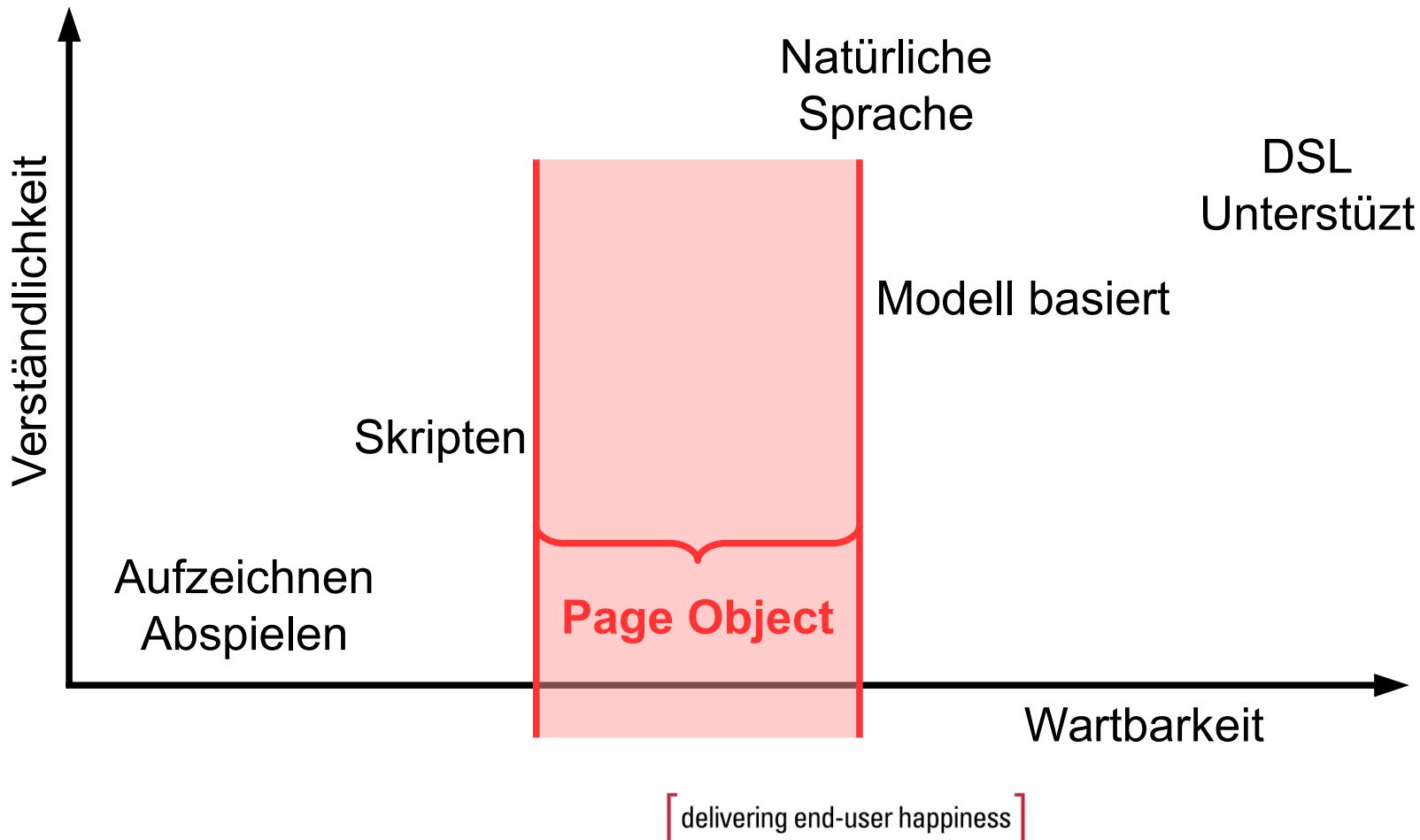
Page Object Pattern

WebDriver & Geb

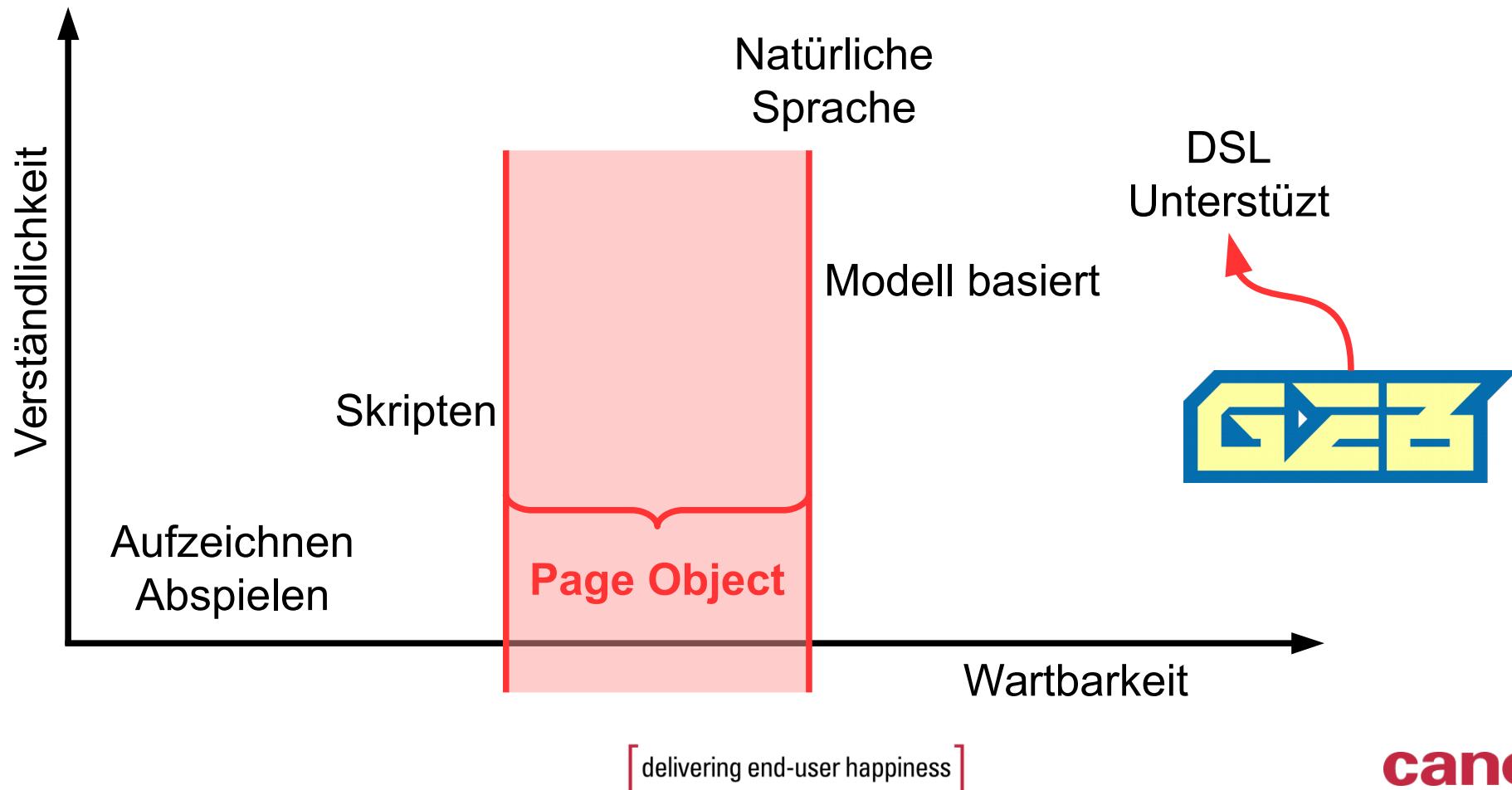
Ansätze in der Übersicht



Ansätze in der Übersicht



Ansätze in der Übersicht



Skript Ansatz

```
import geb.Browser

Browser.drive {
    go "http://gebish.org"

    $("#sidebar .sidemenu a", text: "jQuery-like API").click()

    assert $("#sidebar .sidemenu a", text: "jQuery-like API")
        .parent().hasClass("selected")

    assert $("#main h1")*.text() == ["Navigation", "Form Control"]
}
```

Skript Ansatz

```
import geb.Browser

Browser.drive {
    go "http://gebish.org"

    $("#sidebar .sidemenu a", text: "jQuery-like API").click()

    assert $("#sidebar .sidemenu a", text: "jQuery-like API")
        .parent().hasClass("selected")

    assert $("#main h1")*.text() == ["Navigation", "Form Control"]
}
```

Skript Ansatz

```
import geb.Browser

Browser.drive {
    go "http://gebish.org"

    $("#sidebar .sidemenu a", text: "jQuery-like API").click()

    assert $($(" #sidebar .sidemenu a", text: "jQuery-like API")
        .parent().hasClass("selected"))

    assert $(" #main h1")*.text() == ["Navigation", "Form Control"]
}
```

Skript Ansatz

```
import geb.Browser

Browser.drive {
    go "http://gebish.org"

    $("#sidebar .sidemenu a", text: "jQuery-like API").click()

    assert $("#sidebar .sidemenu a", text: "jQuery-like API")
        .parent().hasClass("selected")

    assert $("#main h1")*.text() == ["Navigation", "Form Control"]
}
```

Page Object Ansatz

```
import geb.Browser

Browser.drive {
    to GebHomePage

    jQueryLikeApi.click()

    assert jQueryLikeApi.selected

    assert sectionTitles == ["Navigation", "Form Control"]
}
```

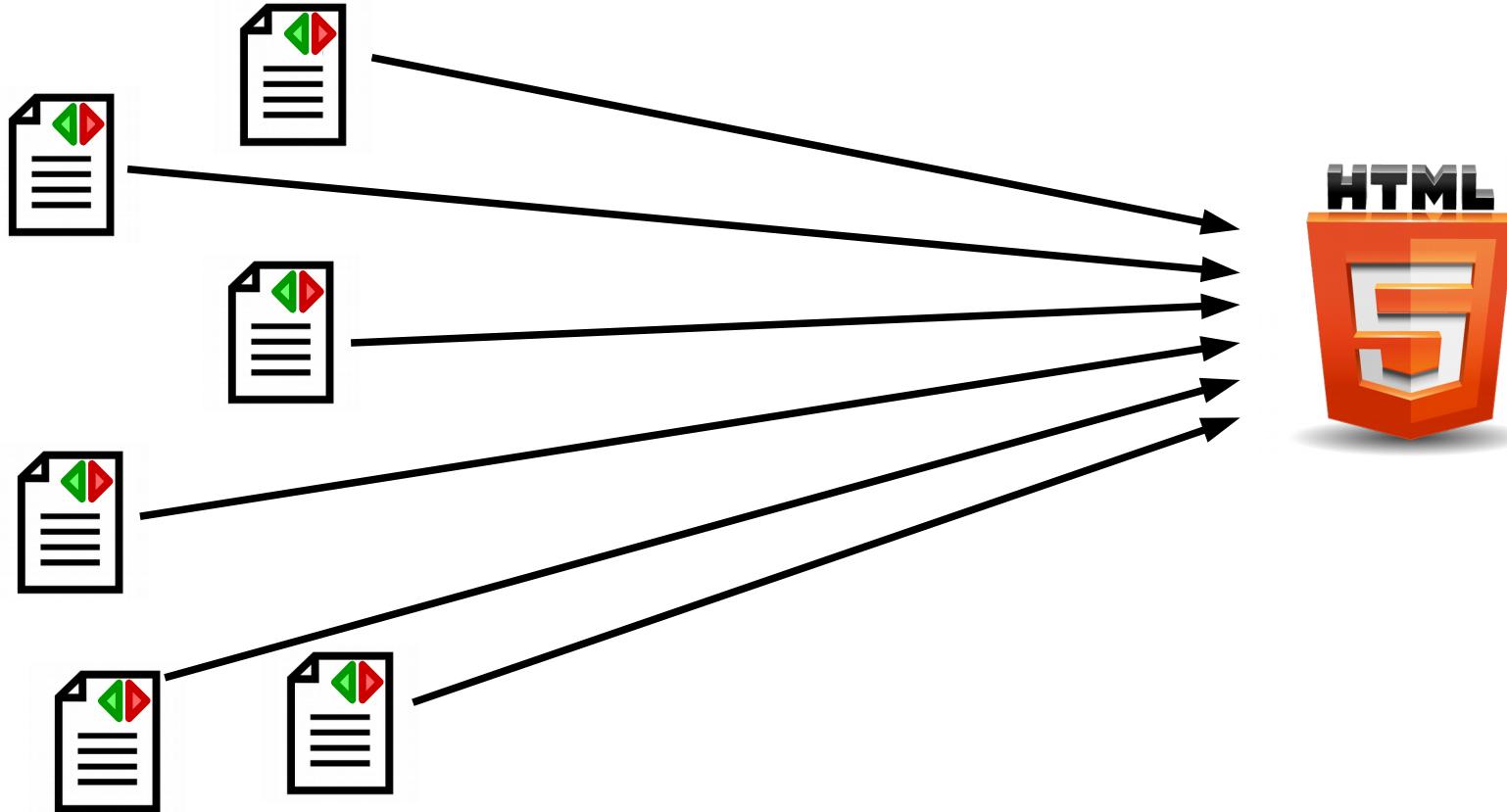
Page Object Ansatz

```
import geb.Browser  
  
Browser.drive {  
    to GebHomePage  
  
    jQueryLikeApi.click()  
  
    assert jQueryLikeApi.selected  
  
    assert sectionTitles == ["Navigation", "Form Control"]  
}
```

GebHomePage

+ jQueryLikeApi
+ sectionTitles

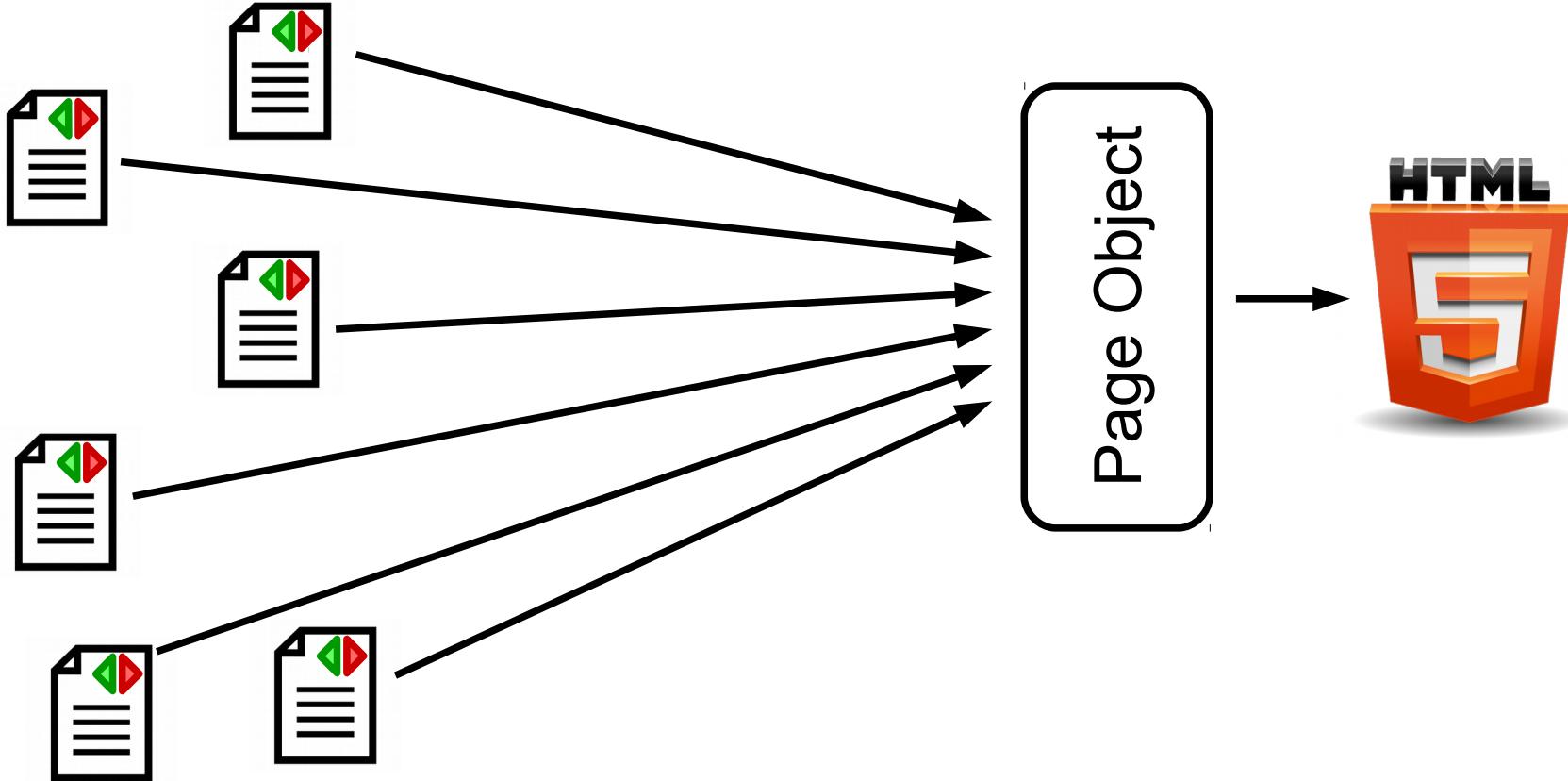
Ohne Page Object



[delivering end-user happiness]

canoo

Mit Page Object

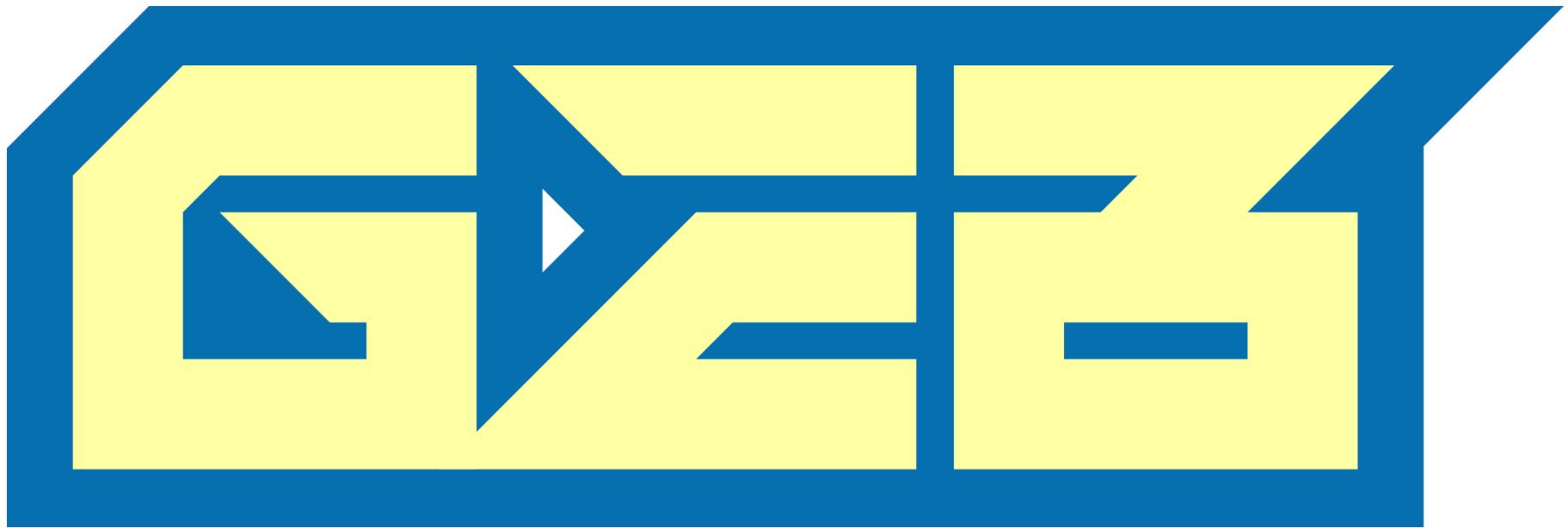


[delivering end-user happiness]

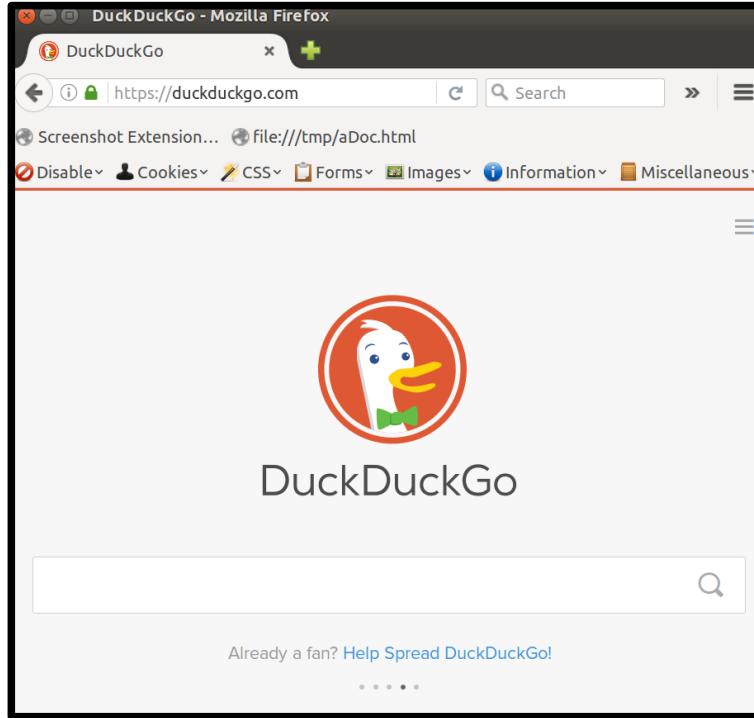
canoo

Page Object – Vorteile

- » Kapselung von Low Level Code
- » Reduktion von Duplizierung
- » Benennung von Elementen



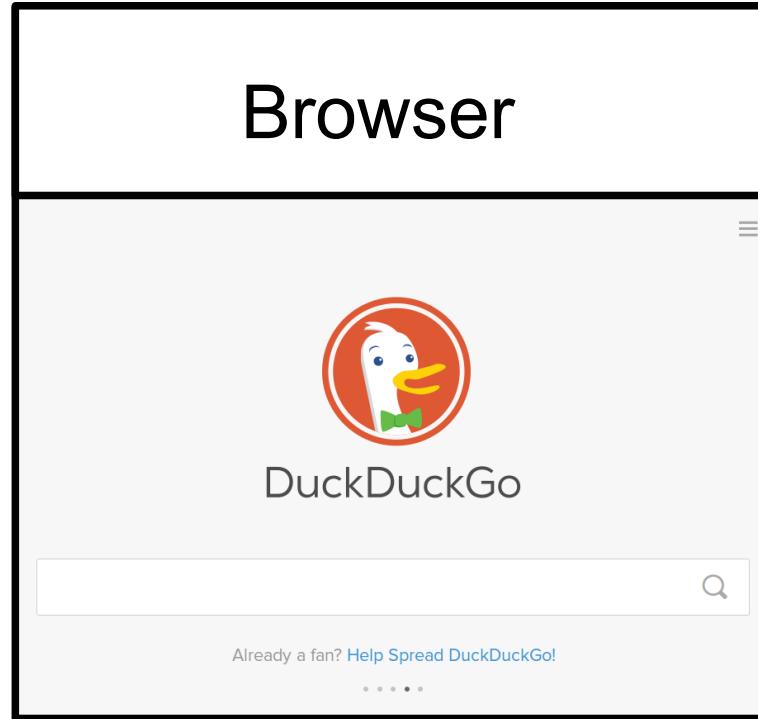
Geb – Architektur



[delivering end-user happiness]

canoo

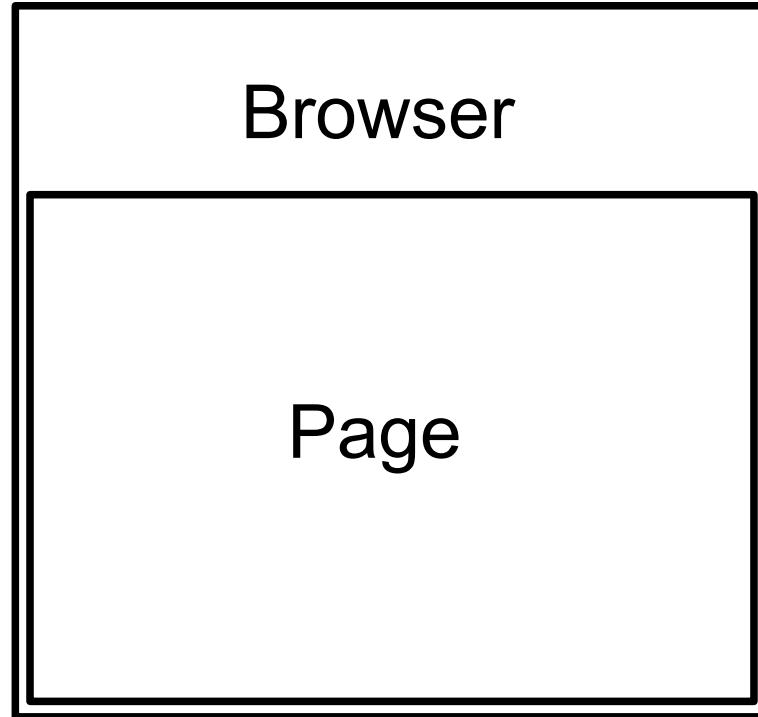
Geb – Architektur



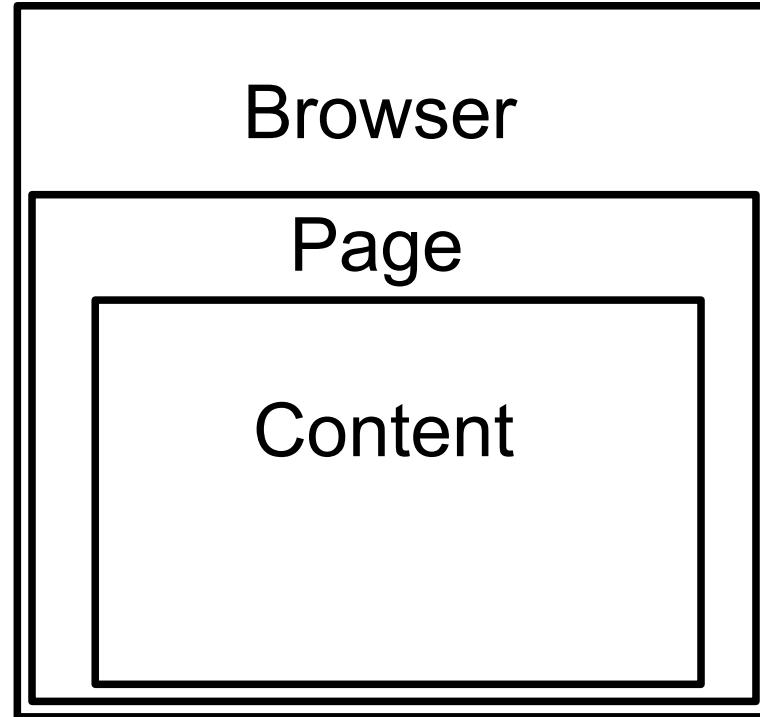
[delivering end-user happiness]

canoo

Geb – Architektur



Geb – Architektur



Geb – Architektur

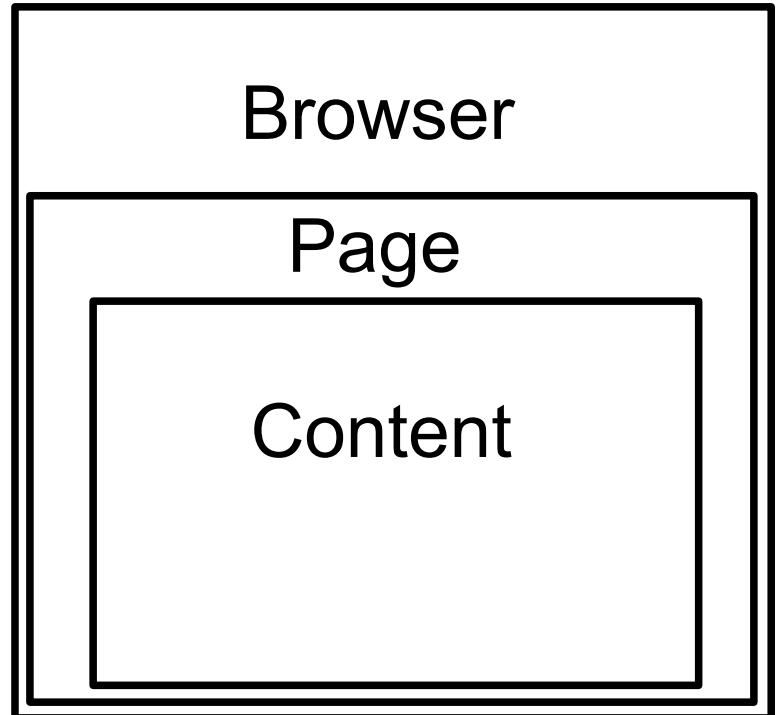
```
import geb.Browser

Browser.drive {
    to GebHomePage

    jQueryLikeApi.click()

    assert jQueryLikeApi.selected

    assert sectionTitles == ["Navigation", "Form Control"]
}
```



Geb – Architektur

```
import geb.Browser

Browser.drive {
    to GebHomePage

    jQueryLikeApi.click()

    assert jQueryLikeApi.selected

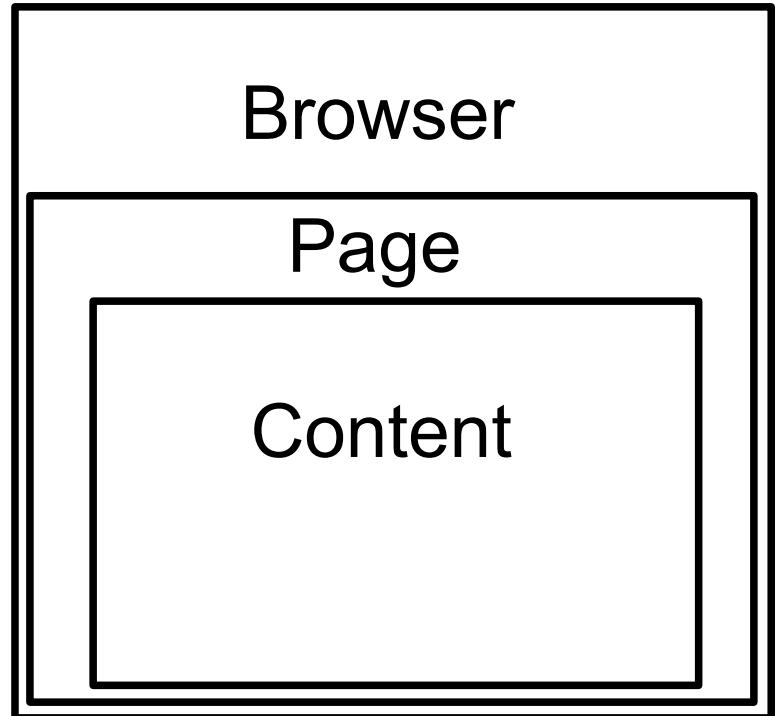
    assert sectionTitles == ["Navigation", "Form Control"]
}
```

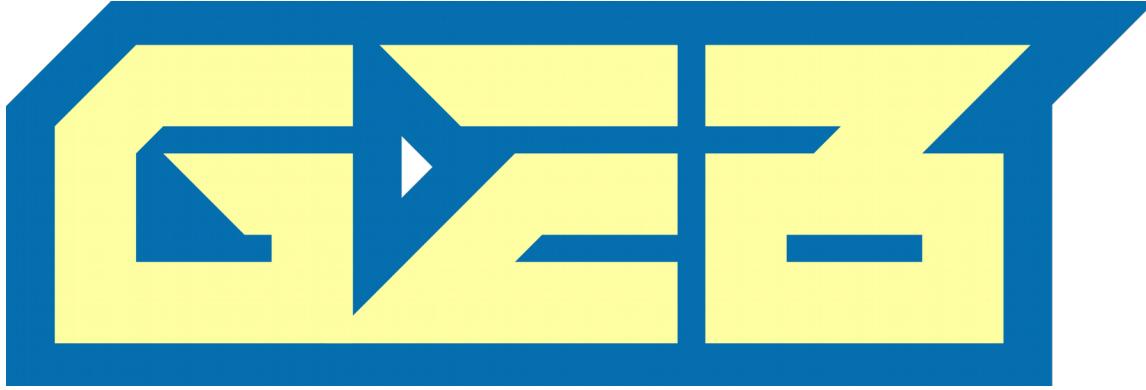
geb

Page

Content

Delegation





Page

Geb – Page

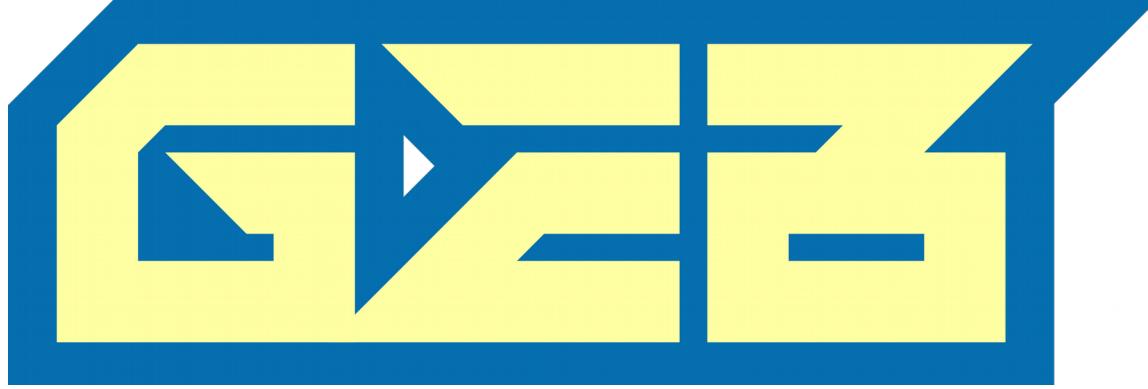
```
class MyPage extends Page {  
    static url = '/foo'  
    static at = { title == 'My Page' }  
    static content = { ... }  
}
```

- » **url:** URL der Seite
- » **at:** Validierung das Browser auf der Seite ist
- » **content:** Elemente innerhalb der Seite

Geb – Page

```
class MyPage extends Page {  
    static content = {  
        header { $('h1', 0) }  
        title { $('div#footer') }  
    }  
}
```

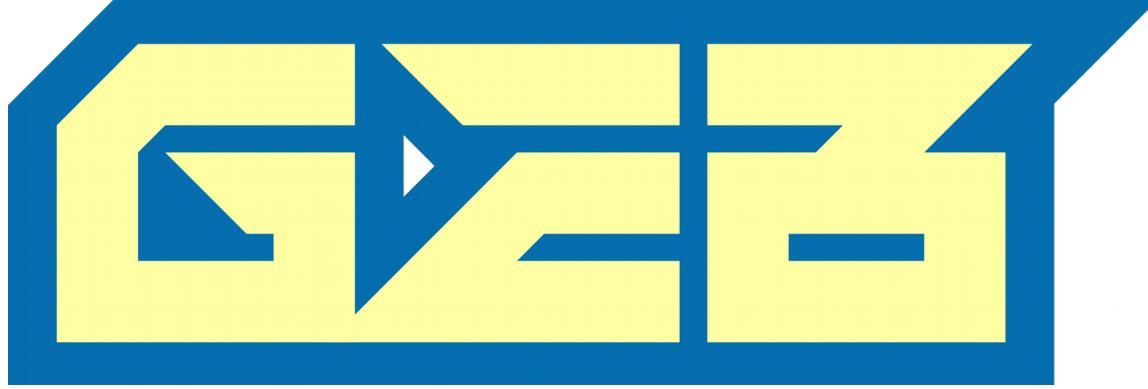
- » Mit dem Content Block wird der Inhalt der Seite modelliert



Content

Geb Content – Hauptaufgaben

- » Selektieren
- » Manipulieren
- » Auslesen



**Content
Selektieren**

Geb – Navigator API

Ein Navigator repräsentiert 0, 1 oder mehrere HTML Elemente

Geb – Navigator API

Ein Navigator repräsentiert 0, 1 oder mehrere
HTML Elemente

0, 1 oder mehrere

Geb – Navigator API – Selektieren

```
$(<css>, <index/range>, <attribute/text>)
```

- » jQuery verwandte Syntax für CSS Selektoren
- » Alle Parameter sind optional
- » Gibt nie NULL zurück

Geb – Navigator API – Selektieren

```
$(<css>, <index/range>, <attribute/text>)
```

- » `$('div.foo')`
// alle <div class="foo"> Elemente
- » `$('div#foo p:first-child[title^="bar"]')`
// Falls vom WebDriver & Browser unterstützt

Geb – Navigator API – Selektieren

```
$(<css>, <index/range>, <attribute/text>)
```

```
» $('div', 0)
  // das erste <div> Element

» $('div', 2..4)
  // das 3., 4. und 5. <div> Element
```

Geb – Navigator API – Selektieren

```
$(<css>, <index/range>, <attribute/text>)
```

```
» $('div', attr: 'foo')
  // alle <div attr="foo"> Elemente

» $('div', text: 'foo')
  // alle <div>foo</div>
```

Geb – Navigator API – Selektieren

```
$(<css>, <index/range>, <attribute/text>)
```

- » `$('div', text: ~/.*foo/)`
// Regexp auf attribute/text
- » `$('div', text: contains('.*foo'))`
// Hilfsmethode erzeugt eine Regexp

Geb – Navigator API – Selektieren

```
// Selektiert eine Teilmenge des Navigator  
» $('.b').filter('div')  
» $('.b').not('div')  
  
» $('div').has('p')  
» $('div').hasNot('p', attr: 'foo')
```

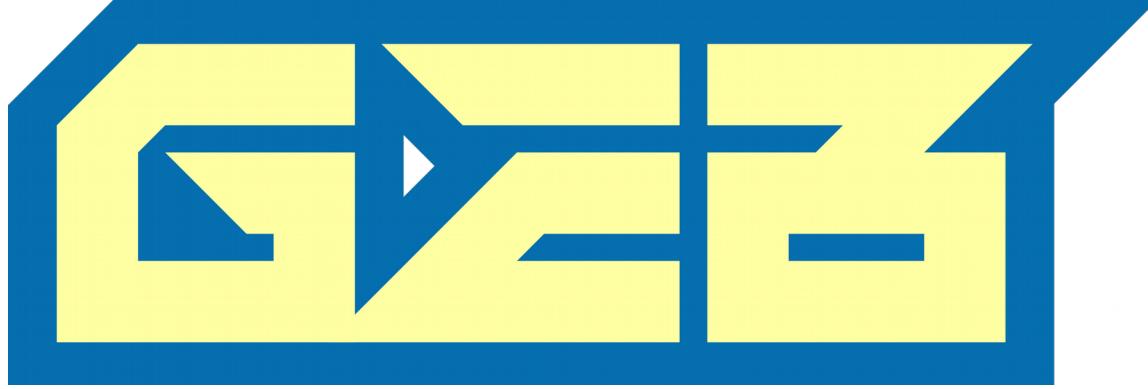
Geb – Navigator API – Selektieren

```
// Rekursiv
» $('#foo').parents('div', attr: 'foo')
» $('#foo').find('form')
```

```
// Nicht Rekursiv
» $('#foo').parent('div')
» $('#foo').children('div')
```

Geb – Navigator API – Selektieren

```
// Geschwister / Nachbarn  
» $('#foo').previous('div')  
» $('#foo').prevAll('div')  
  
» $('#foo').next('div')  
» $('#foo').nextAll('div')  
  
» $('#foo').siblings('div')
```



**Content
Manipulieren**

Geb – Navigator API – Manipulieren

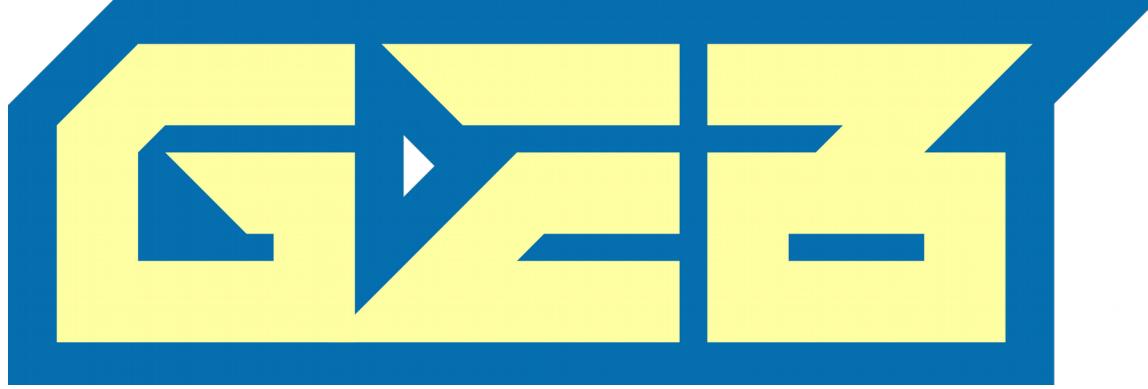
```
» $('#foo').click()  
// einfach immer drauf klicken
```

Geb – Navigator API – Manipulieren

```
» $('input#username').value('test')
» $('input#username') = 'test'
  // Schreiben (Ersetzen)

» $('input#username') << 'user'
  // Schreiben (Anfügen)

» $('input#pwd') << Keys.chord(Keys.CONTROL, 'v')
  // Auch Tastenkombinationen möglich
```



Content Auslesen

Geb – Navigator API – Auslesen

```
» $('#foo').isDisplayed()  
// ist das Element sichtbar
```

Geb – Navigator API – Auslesen

```
<p id="a" class="b c" foo="d">Inhalt</p>
```

- » `$('#a').text()` == 'Inhalt'
- » `$('#a').tag()` == 'p'
- » `$('#a').classes()` == ['b' , 'c']
- » `$('#a').attr('foo')` == 'd'
- » `$('#a').@foo()` == 'd'

Geb – Navigator API – Auslesen

```
» $('input#bar').value()  
  
» $('input#bar').isEnabled()  
» $('input#bar').isDisabled()  
  // Zustand des @disabled Attribut  
  
» $('input#bar').isEditable()  
» $('input#bar').isReadOnly()  
  // Zustand des @readonly Attribut
```

GEB – Vorteile

- » Groovy unterstütze DSL und Delegation
- » jQuery ähnliche Syntax
- » Starke Integration des Page Object Pattern
- » Power Asserts



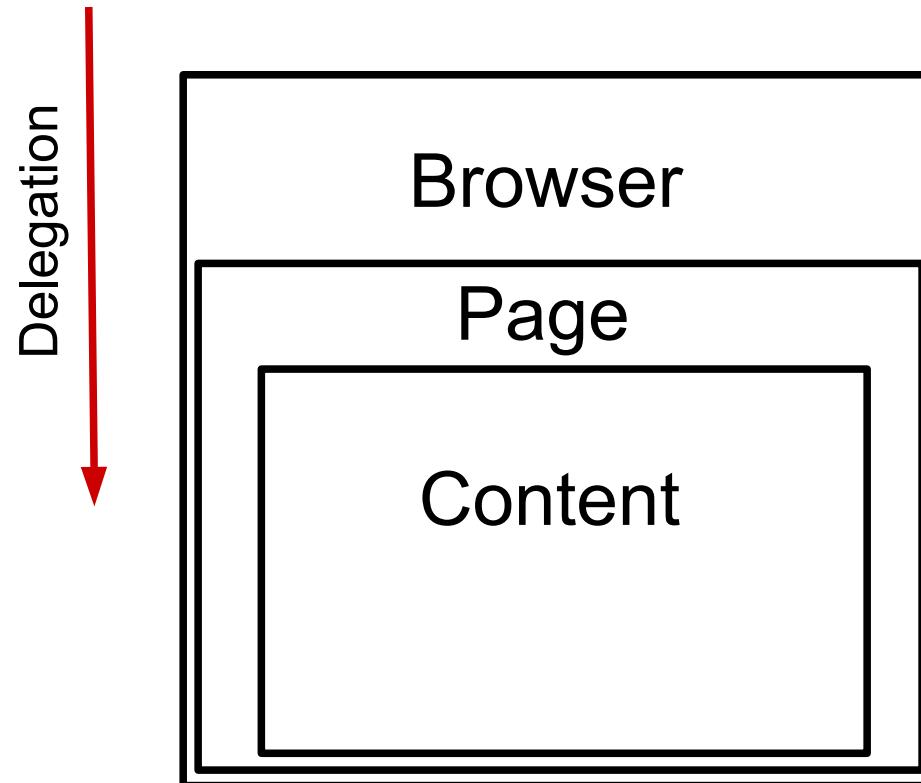
Spock – Beispiel

```
class MySpec extends Specification {  
    def "test something"() {  
        given:  
            int a = 2  
        when:  
            int result = 2 * a  
        then:  
            result == 4  
    }  
}
```

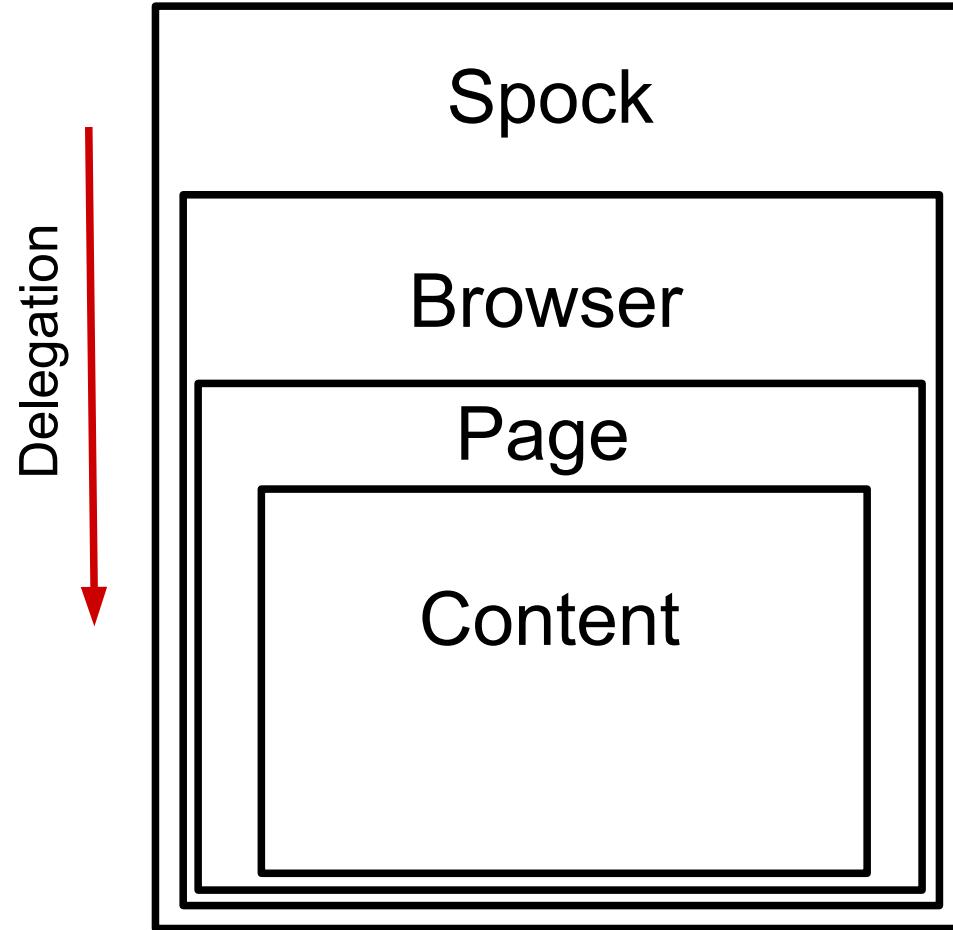
Spock – Beispiel

```
class MySpec extends Specification {  
  
    void setup() { ... }      // @Before  
    void setupSpec() { ... } // @BeforeClass  
  
    void cleanup() { ... }    // @After  
    void cleanupSpec() { ... } // @AfterClass  
}
```

Geb & Spock



Geb & Spock



[delivering end-user happiness]

canoo

Spock – Beispiel

```
@Stepwise
class MySpec extends GebReportingSpec {

    def 'smoke test'() {
        when:
            go 'localhost:8080/foo'

        then:
            title == 'Hello World'
    }
}
```

Spock – Vorteile

- » Groovy Delegation
- » @Stepwise
- » ReportingSpec
- » Power Asserts



Single Page Application

Synchron vs. Asynchron

- » Bei klassischem Laden einer Seite weiss der Browser wann die Seite geladen ist.
- » Bei asynchronem (Nach-)Laden von Daten und Elementen weiss der Browser nicht wann die Seite geladen ist.

Warten, warten, warten

- » Warten bis Seite “geladen“ ist – Geb bietet Hilfe

```
$('a#loadMore').click()  
  
waitFor {  
    items.size() > 10  
}  
  
items[10].click()
```

Aber worauf soll man warten?

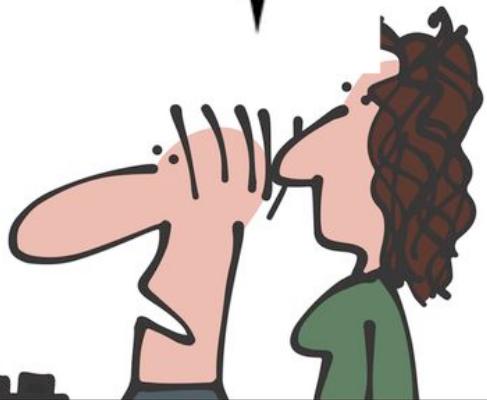
- » Block UI
 - Elemente werden hinter dem Overlay geladen
- » Alle Request fertig
 - Counter im JS verwenden

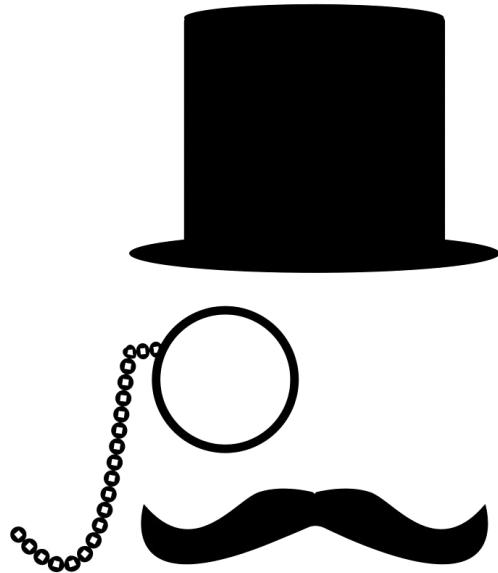
**TIPS &
TRICKS**

Readability



PLEASE DON'T ASK ME
HOW THIS WORKS AND
PLEASE DON'T ASK ME
WHY THIS WORKS!!





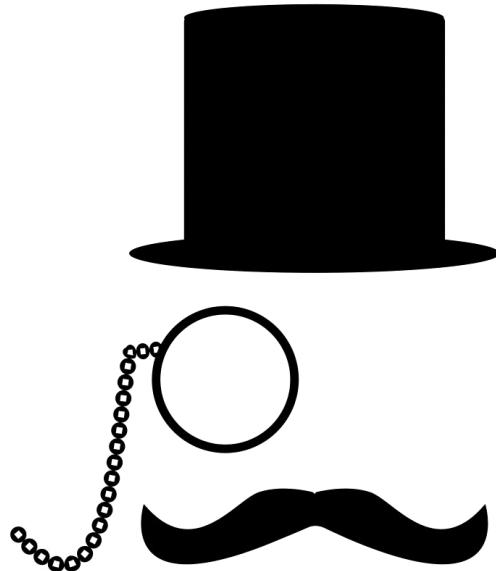
Tipp 1

[delivering end-user happiness]

canoo

HTML Markieren

- » CSS-Klassen verwenden um relevante Elemente zu markieren
 - IDs sind oft ungeeignet da unique
 - Prefix um zu markieren dass die CSS-Klasse nicht fürs Styling verwendet wird



Tipp 2

[delivering end-user happiness]

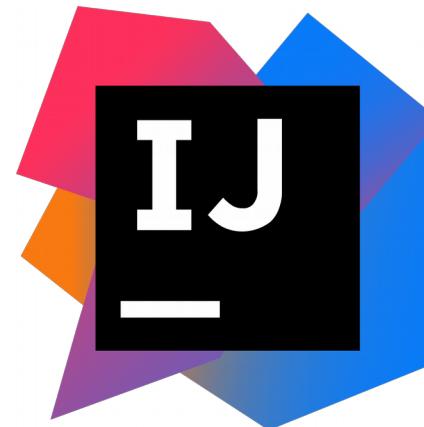
canoo

Page explizit machen

- » Geb und Spock sind Groovy
 - » Groovy ist eine dynamische Sprache
 - » IDE ist meist ein wenig überfordert
-
- » Delegation
 - Spock → Browser → Page

Page explizit machen

- » Geb und Spock sind Groovy
 - » Groovy ist eine dynamische Sprache
 - » IDE ist meist ein wenig überfordert
-
- » Delegation
 - Spock → Browser → Page



canoo

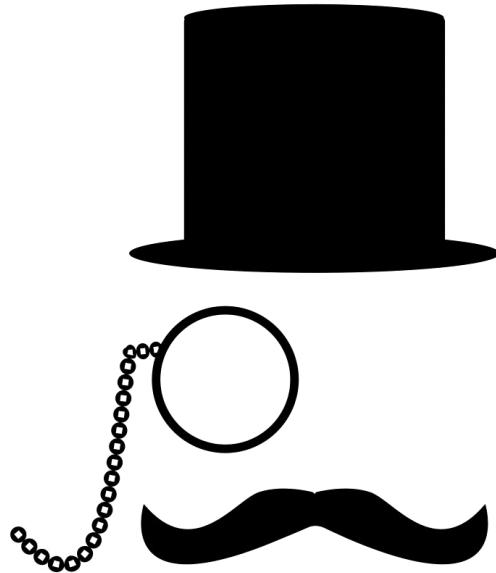
Page explizit machen

```
def 'someTest'() {  
    given:  
        at MainPage  
    when:  
        aboutPageLink.click()  
    then:  
        at AboutPage  
        version == '1.2.6'  
}
```

Page explizit machen

```
@Shared Page p
```

```
def 'someTest'() {
    given:
        p = at MainPage
    when:
        p.aboutPageLink.click()
        p = at AboutPage
    then:
        p.version == '1.2.6'
}
```



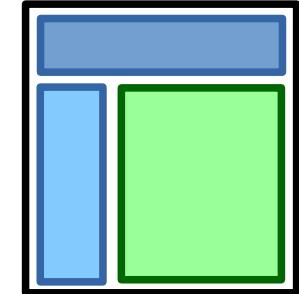
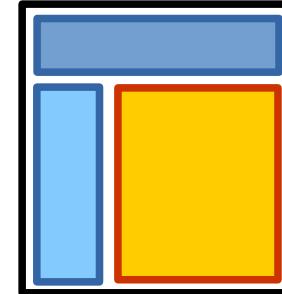
Tipp 3

[delivering end-user happiness]

canoo

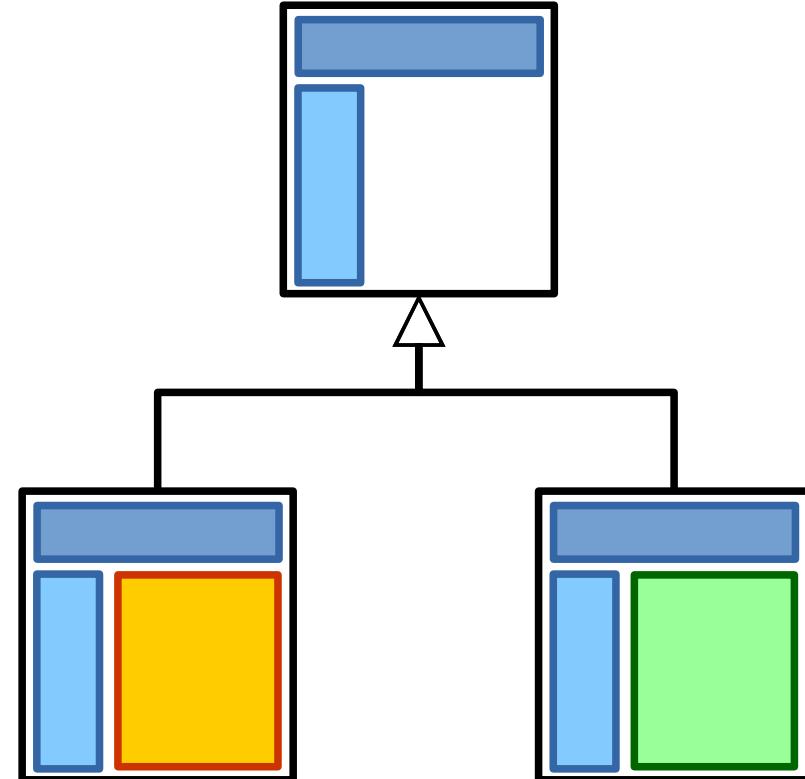
Geb – Page

» Der Content wird vererbt

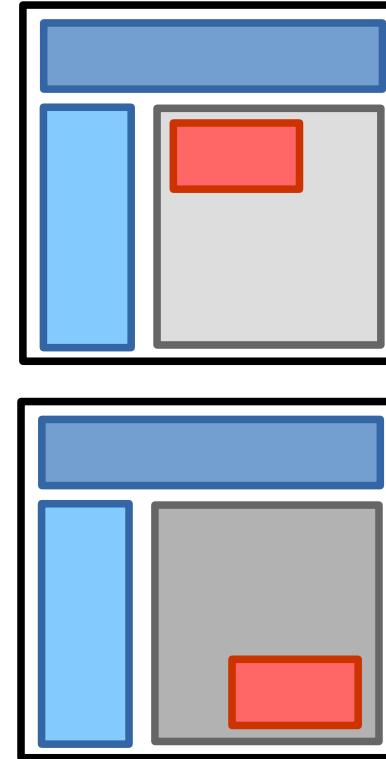
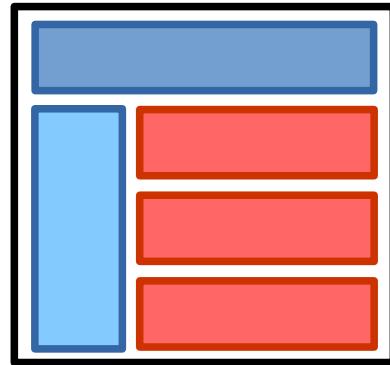


Geb – Page

- » Der Content wird vererbt



Geb – Module



[delivering end-user happiness]

canoo

Geb – Module

```
class MyModule extends Module {  
    static content = {  
        name { $('td', 0) }  
        price { $('td', 1) }  
    }  
}
```

- » Gruppert mehrere Elemente
- » Wie eine Page aber ohne URL

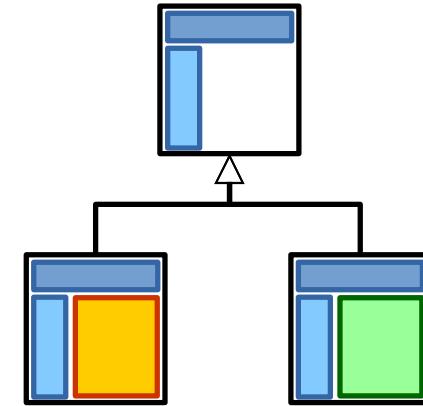
Geb – Module

```
static content {  
    shoppingCart { $('#cart').module(Cart) }  
    searchHits { $('tr').moduleList(SearchHit) }  
}
```

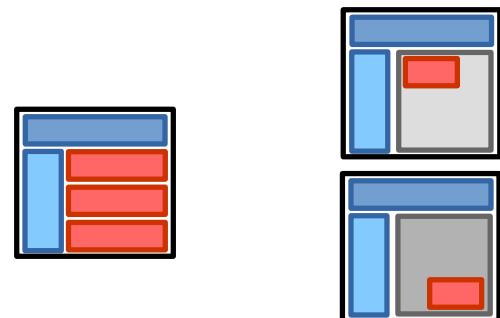
```
shopingCart.totalPrice  
searchHits[8].title
```

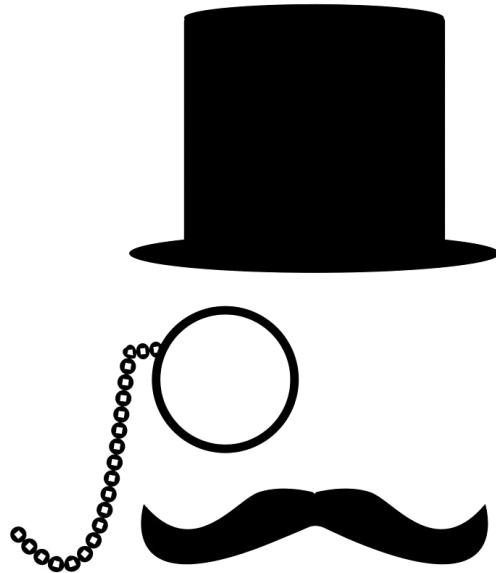
Strukturiert Code

- » Basisklassen für Tests und Pages
 - Für Code-reuse in Tests OK
 - Dennoch nicht übertreiben



- » Geb Module für Listen und Pop-Ups



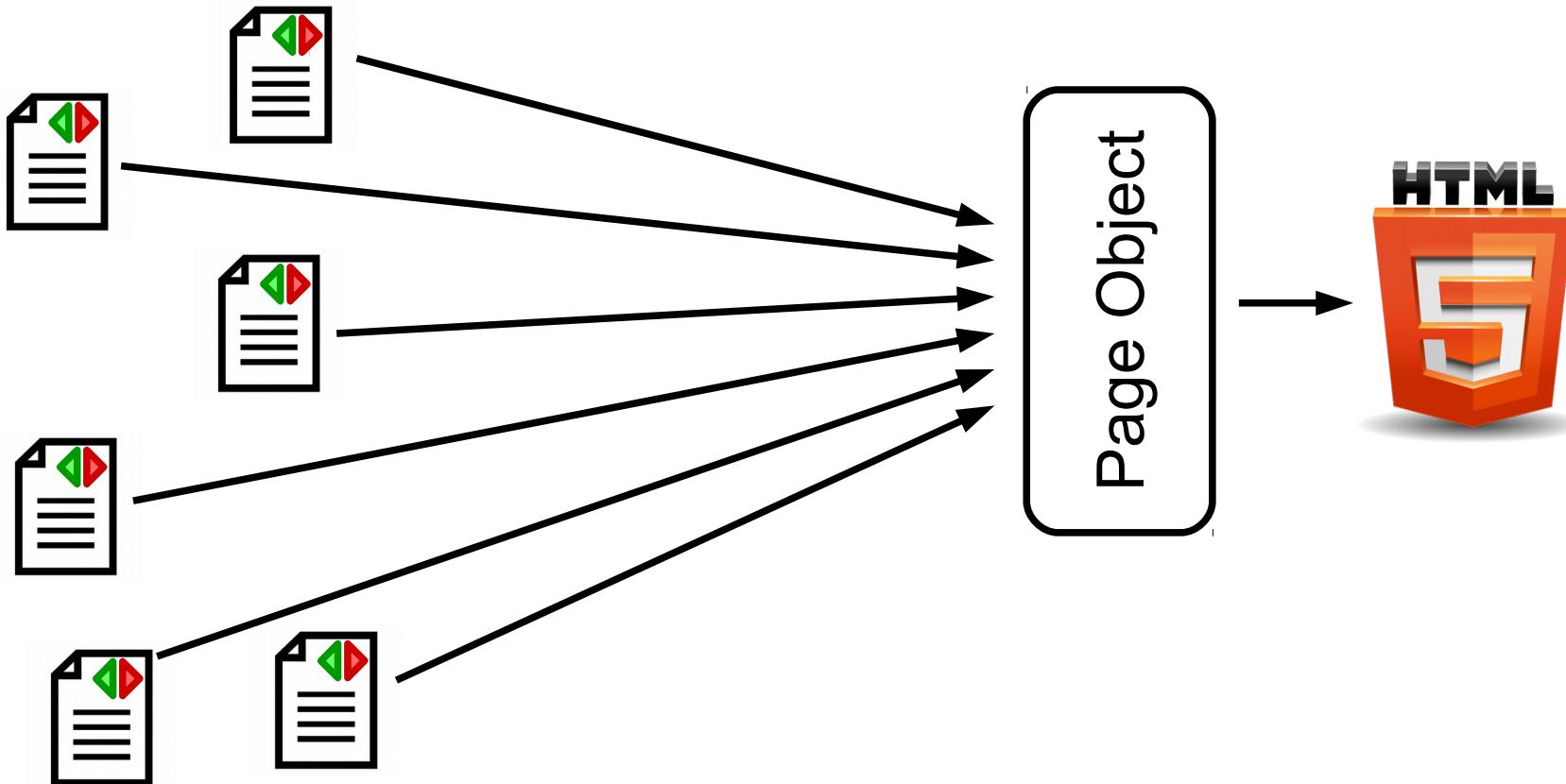


Tipp 4

[delivering end-user happiness]

canoo

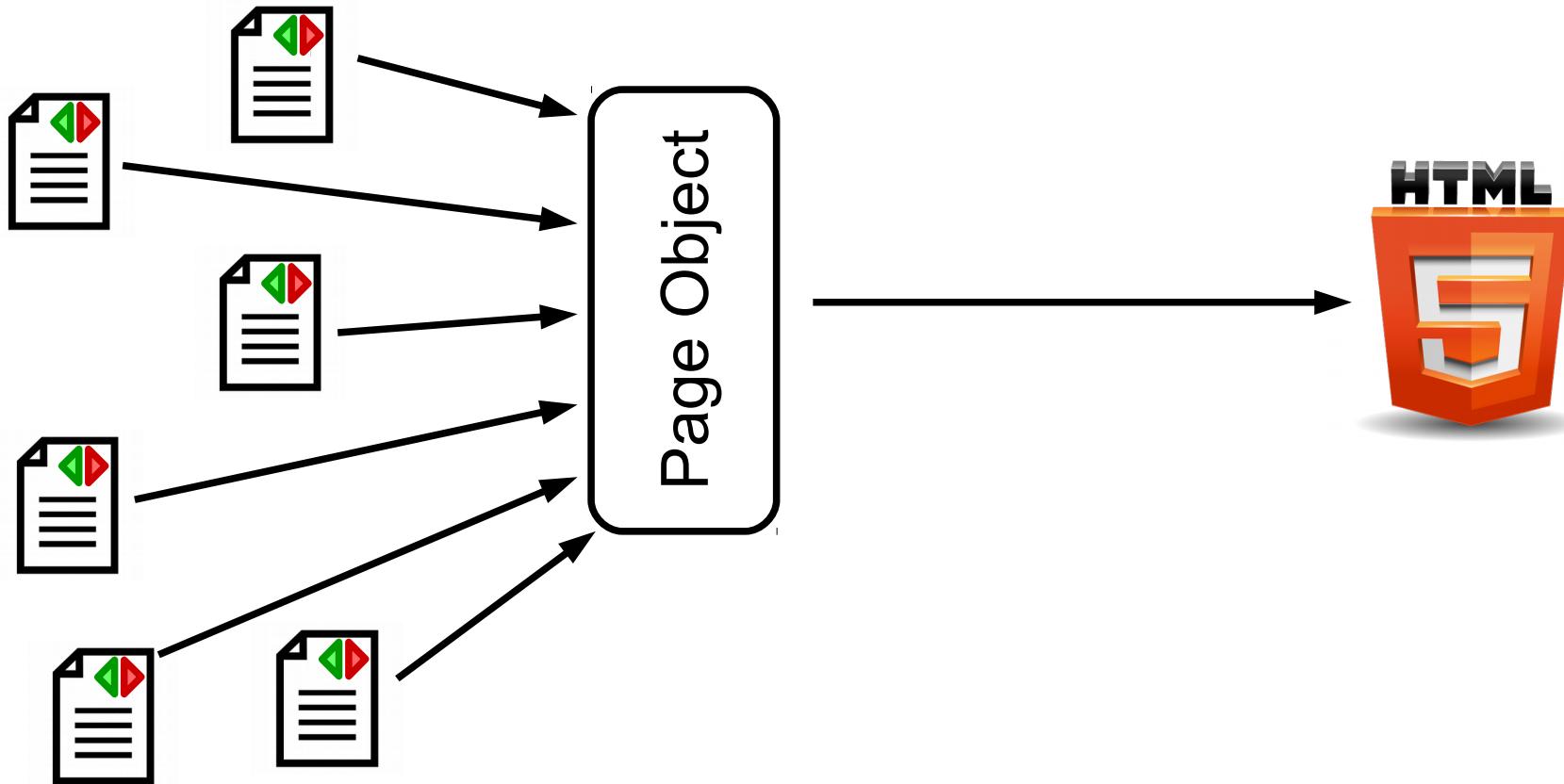
Page Object Design



[delivering end-user happiness]

canoo

Page Object Design



delivering end-user happiness

canoo

Geb – Page

```
class MyPage extends Page {  
    static content = {  
        header { $('h1', 0) }  
        title { $('div#bar') }  
    }  
}
```

- » Mit dem Content Block wird der Inhalt der Seite modelliert

Geb – Page

```
class MyPage extends Page {  
    static content = {  
        itemCount { $('.item').size() }  
        name { $('input#name').value() }  
    }  
}
```

Tipp:

- » Muss kein Navigator Objekt sein !!

Geb – Page

```
class MyPage extends Page {  
    static content = {  
        foo { String arg ->  
            $('div', title: arg) }  
    }  
}
```

Tipp:

- » Es können Argumente übergeben werden !!

Geb – Page

```
class MyPage extends Page {  
    static content = { ... }  
    void foo() {  
        // kann content verwenden  
    }  
}
```

Tipp:

- » Normal Methoden sind auch möglich

Page Object näher an den Test

- » Content der Page soll dem Test die Information in der richtigen Form liefern
 - Meistens keine Navigatoren
 - Parametrierter Content
- » Benennt was der Benutzer auf der Page sieht

Page Object näher an den Test

```
p.titleLabel.text().contains(user.name)
```

» Name im PageObject extrahieren

```
p.userName == user.name
```

Page Object näher an den Test

```
p.tableRows[8].text().contains('foo')
```

» Parametrierter Content

```
p.product(8) == 'foo'
```

Page Object näher an den Test

- » Methoden für Aktionen
 - Navigieren
 - Ausfüllen einfacher Formulare
 - Eine Methode pro Ausgang/Resultat
- » Benennt was der Benutzer auf der Page macht

Page Object näher an den Test

```
p.aboutPageLink.click()  
p = waitFor { at AboutPage }
```

» Aktions Methode auf der Page







Langsame Tests

[delivering end-user happiness]

canoo



Parallelisieren

- » Spock kann mehrere Tests parallel ausführen
- » Gruppiert Tests und lässt mehrere Gruppen parallel von eurem CI Server(n) ausführen



Falsche Daten

[delivering end-user happiness]

canoo

Daten Management

- » Erzeuge die Daten für den Test:
 - In der setupSpec() Methode
 - Räume in der cleanupSpec() Methode auf
 - Verwende unique Strings (Zeit/Hash)
 - Builder Pattern



Login schlägt fehl

[delivering end-user happiness]

canoo



**KEEP
CALM
AND
LOGOUT AND
LOG BACK IN**

[delivering end-user happiness]

canoo

Login schlägt fehl

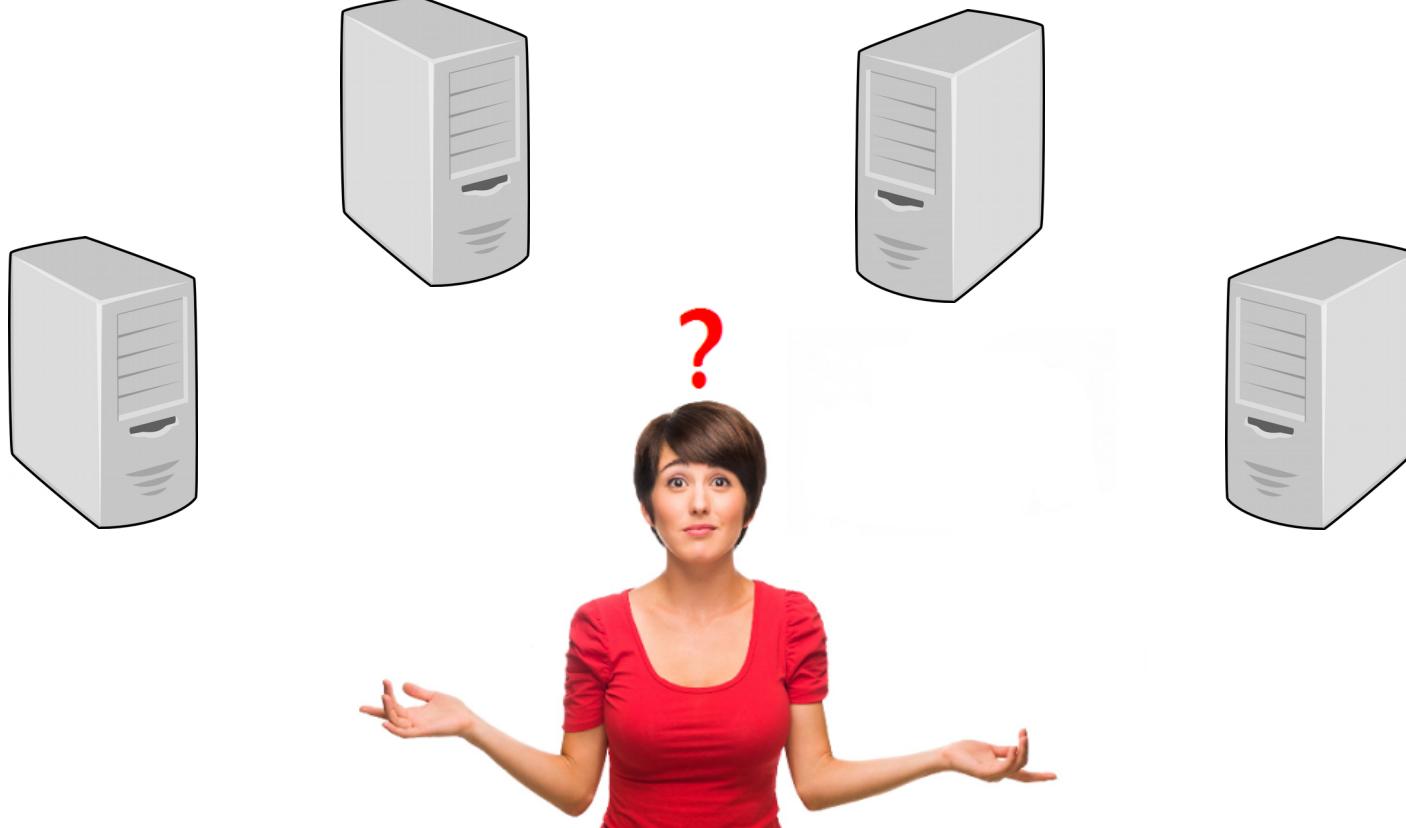
- » Am Ende jedes Tests ausloggen
 - In der cleanupSpec() Methode
 - In einer Basisklasse



Works on my Machine

[delivering end-user happiness]

canoo



[delivering end-user happiness]

canoo

Works on my machine

- » Laufzeitinfo in der Fusszeile anzeigen
 - Server / DB Name
 - Version (inkl. Commit-ID)

Schreibt UI-Tests mit Geb und Spock

- » <http://www.gebish.org>
- » <http://spockframework.org>