



Legacy Code meistern in x einfachen Schritten

und testgetrieben weiterentwickeln

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.0



Abstract (Teil 1)

In einer idealen Welt würden wir nur "neuen" Code schreiben, der natürlich perfekt und wunderschön ist. Wir müssten nie wieder unseren Code anschauen, geschweige denn 10 Jahre alte Projekte warten. Ende des Tagtraums ... Leider ist unsere Welt nicht so ideal, unser Code von gestern ist heute schon Legacy. Diesen im Nachhinein zu verstehen, zu erweitern oder darin Fehler zu beheben, ist immer eine Herausforderung, insbesondere wenn Tests fehlen.

Trotzdem gibt es einfache Möglichkeiten, wie man die Qualität von Legacy-Code verbessern kann. Das Wichtigste ist das Einziehen von Fangnetzen, sodass man trotz fehlender Tests guten Gewissens Änderungen durchführen kann. Wer Golden Master, Subclass to Test und Extract Pure Functions an konkreten Beispielen kennenlernen möchte, ist in dieser Session genau richtig.



Bestandsanwendungen müssen gewartet und ggf. weiterentwickelt werden, bergen aber meist viele Defekte. Als Entwickler fürchten wir uns zudem, mehr Schaden anzurichten, weil das Verständnis für den Legacy Code fehlt. Refactoring kann zum Verstehen beitragen, endet aber aufgrund der typischerweise fehlenden automatisierten Tests in einem Blindflug.

Mit testgetriebener Entwicklung scheint es eine Allzweckwaffe für gutes Design und eine geringe Fehlerrate zu geben. Aber TDD und Legacy Code scheinen sich eigentlich auszuschließen. Anhand von Live Coding schauen wir, wie die testgetriebene Entwicklung trotzdem helfen kann, den Code ohne allzu große Bauchschmerzen anzupassen. Schöner Nebeneffekt wird das Entstehen eines automatisierten Testbetts sein, welches zukünftige Refactorings leichter machen wird.

Falk Sippach (@sipsack)

Trainer, Berater, Entwickler



Co-Organisator



Schwerpunkte
Architektur
Agile Softwareentwicklung
Codequalität

Java und XML

) Software Factory)

- Schlüsselfertige Realisierung von Java Software
- Individualsoftware
- Pilot- und Migrationsprojekte
- Sanierung von Software
- Software Wartung

) Object Rangers)

- Unterstützung laufender Java Projekte
- Perfect Match
- Rent-a-team
- Coaching on the project
- Inhouse Outsourcing

) Competence Center)

- Schulungen, Coaching, Weiterbildungsberatung, Train & Solve-Programme
- Methoden, Standards und Tools für die Entwicklung von offenen, unternehmensweiten Systemen

Unser Thema heute:

Refactoring Legacy Code + TDLCR

Disclaimer: KEIN Projekterfahrungsbericht

Live-Coding



Geertjan Wielenga @GeertjanW · 8. Nov.
How much better would demos be if attendees would come to the speaker's hotel room, where demos always work perfectly. Plus, the minibar!

Michael Simons @rotnroll666 · 8. Nov.
@GeertjanW like a minibarcamp? 🍷



Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0 Public Domain Lizenz) <https://twitter.com/GeertjanW/status/796021888745144320>

Hugo

Kenn'se noch den Meier?

Genau, der ist in Rente gegangen.

Brauche aber ein neues Feature!

Dieter

Der hat doch die alte Telefon-Tarif-Anwendung gewartet.

Orientation in Objects

Wir wollen den Kunden Rabatte gewähren können.

Bauen Sie mir das bis **nächste Woche** ein!

Ähm, in 3 Tagen?





Hugo

Dieter

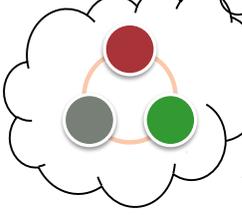
Orientation in Objects

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 9

Und bitte nichts kaputt machen, das ist eine **unternehmenskritische Anwendung!**



Hugo

Dieter

Orientation in Objects

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 10

TDLCR

Test Driven Development

TDD hilft beim Verbessern des Codes

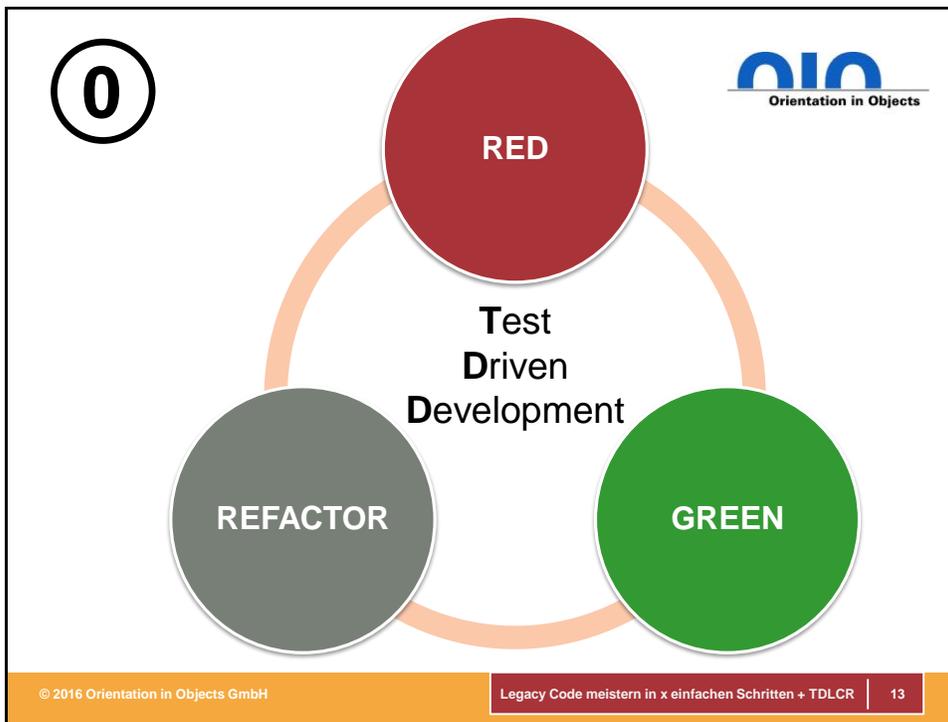


TDD is not a design tool. It's a software development workflow that has prompts for code improvement in its lifecycle #TDD #programming

Übersetzung anzeigen



<https://mobile.twitter.com/TDDQuotes/status/783787123237265408>



"Gesetze" von TDD

Orientation in Objects

- Schreibe einen fehlschlagenden Test, bevor du Code für das Produktivsystem verfasst.
- Schreibe nur so viel Code für den Test, damit er kompiliert (**Rot**)
- Schreibe nur so viel Code, um den Test zu bestehen (**Grün**)
- Refactor: Duplikation entfernen

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 14

Live-Coding

Rabatt-Klasse mit Prozentsatz

10 % Rabatt => berechnePreis liefert 90 % vom Preis zurück

kein Rabatt => berechnePreis liefert gleichen Wert zurück

ignoriere negative Rabatte => berechnePreis liefert gleichen Wert zurück

Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0 Public Domain Lizenz)

Tipps TDD

- kleine Schritte
- Workflow (Red – Green - Refactor) einhalten
- Ideen für weitere Testfälle in eine Liste schreiben und nach und nach abarbeiten (kleine Schritte!)

Vorteile TDD

- kein ungetesteter Code => sehr hohe Testabdeckung
- saubere/testbare Architektur durch TDD als Designstrategie
 - Anforderungen werden durch Tests dokumentiert
 - Software weniger fehleranfällig und wartungsintensiv
- hohe Qualität und verständlicher Code durch ständiges Aufräumen
 - Entfernen von Duplikationen
 - Abstrahieren wo sinnvoll
 - Einhalten der Code-Konventionen
- kein unnötiger Code auf Vorrat (YAGNI)
 - Konzentration auf das Wesentliche

TDLCR

Legacy Code

Legacy Code meistern in **8** einfachen Schritten

Nur heute,
nicht 1,
nicht 2,
...



LEGACY

Vermächtnis

Erbe

Altlast

Hinterlassenschaft

Foto von smpcas, [CC0 Public Domain Lizenz, https://pixabay.com/de/pula-kroatien-amphitheater-erbe-827909/](https://pixabay.com/de/pula-kroatien-amphitheater-erbe-827909/)

Greenfield, Agile, TDD, ...

`// TODO: refactor this`

Foto von Antranias, CC0 Public Domain Lizenz, <https://pixabay.com/de/bauarbeiten-ger%C3%BCst-baustelle-670278/>

Foto von John Hult, CC0 1.0 Universal, <https://unsplash.com/search/ruin?photo=uh0u8OH4zuE>

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 21

Was ist Legacy Code?

Alter Code, geerbter Code, Code den niemand mehr versteht ...

"Some crap made by someone else"

Code entstanden unter Zeitdruck und ohne automatisierte Tests

Technische Schulden

Was ist mit unserem eigenen Code?

Foto von John Hult, CC0 1.0 Universal, <https://unsplash.com/search/ruin?photo=uh0u8OH4zuE>

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 22

Warum Legacy Code anfassen?

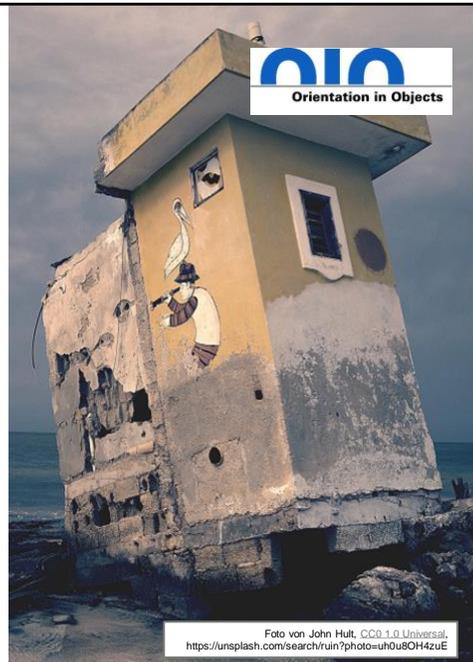
Verstehen

Erweitern

Bugfixing

Optimierung

© 2016 Orientation in Objects GmbH



Legacy Code meistern in x einfachen Schritten + TDLCR 23

Wo kann TDD bei Legacy Code helfen?

~~Verstehen~~

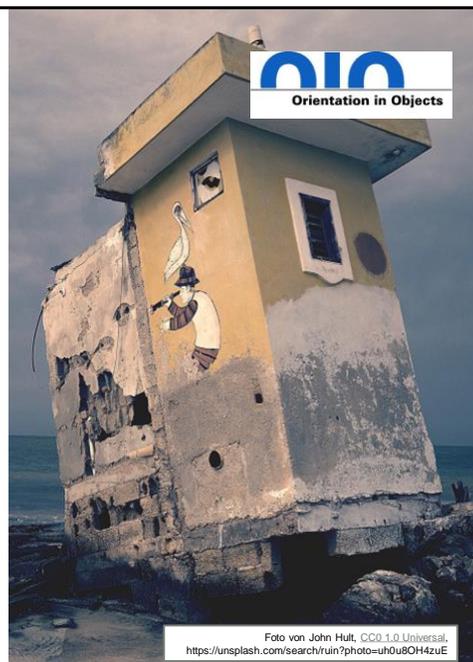
Erweitern

~~Bugfixing~~

~~Optimierung~~



© 2016 Orientation in Objects GmbH



Legacy Code meistern in x einfachen Schritten + TDLCR 24

TDD vs. Legacy Code

**Test vor dem Code vs.
Code schon da**

TDLCR

Test Driven Legacy Code Refactoring



oio
Orientation in Objects

**Erweitern Strategie 1:
Edit and Pray!**

Foto von Myriams-Fotos: <https://pixabay.com/en/rosary-faith-pray-folded-hands-1211064/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 27



oio
Orientation in Objects

**Erweitern Strategie 2:
Canary in the
coal mine**

Foto von janjf93: <https://pixabay.com/en/canary-bird-warrant-canary-road-1606695/> (CC0 Public Domain Lizenz)

Legacy Code nio
Orientation in Objects

Somebody else's code

Was ist mit unserem eigenen Code?

Jeder kennt ihn ...

Keiner mag ihn ...

© 2016 Orientation in Objects GmbH Legacy Code meistern in x einfachen Schritten + TDLCR | 29

Refactoring Legacy Code nio
Orientation in Objects

Warum?

Verstehen

Erweitern

Bugfixing

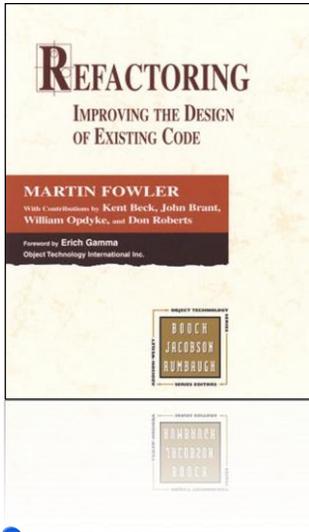
Optimierung



Grafik von CtkerFreeVectorImages: <https://pixabay.com/de/vitamine-tabletten-pillen-medizin-26622/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH Legacy Code meistern in x einfachen Schritten + TDLCR | 30

Refactoring ~~Legacy~~ Code



Annahmen

Es gibt automatisierte Tests ...

Quellcode ist schon testbar ...

Refactoring ~~Legacy~~ Code

Code Smells 

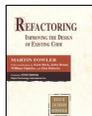
Temporary Field

Long Method

Feature Envy

...

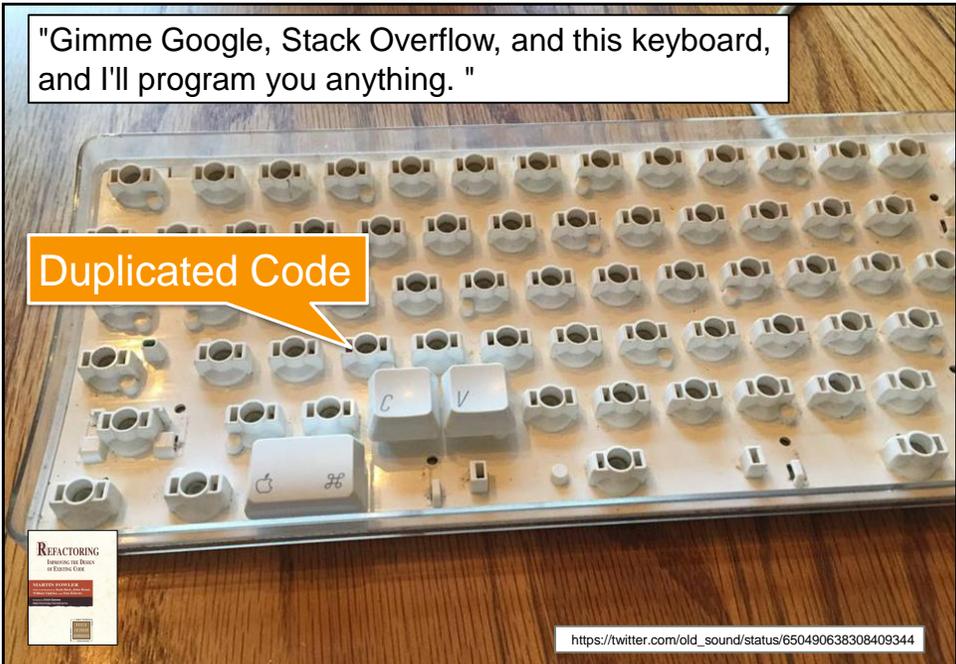
Code Comments



<https://twitter.com/petecheslock/status/646507209413775360/photo/1>

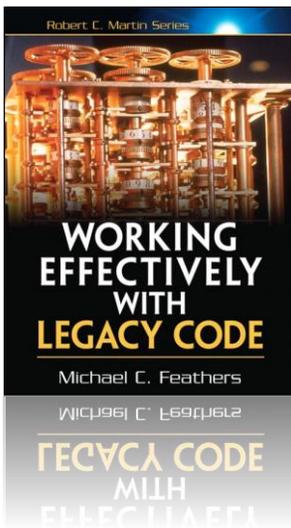
"Gimme Google, Stack Overflow, and this keyboard, and I'll program you anything. "

Duplicated Code



https://twitter.com/old_sound/status/650490638308409344

Refactoring Legacy Code



**“ Code
without
tests**



“ Code
without tests
is bad code.

Michael Feathers



J. B. Rainsberger

“ Legacy code is
valuable code
that we feel afraid
to change.



Foto von PublicDomainPictures, [CC0 Public Domain Lizenz](https://pixabay.com/de/menschen-abdeckung-schrei-314481/), <https://pixabay.com/de/menschen-abdeckung-schrei-314481/>

Es ist egal, wie ...

... gut geschrieben der Code ist

... schön der Code ist

... objektorientiert der Code ist

... entkoppelt der Code ist

Tests

lassen unser Verhalten schnell und verifizierbar ändern

Ohne Tests

wissen wir nicht, ob der Code besser oder schlechter wird

Die gute Nachricht ...

Keine Wissenschaft

Gesunder Menschenverstand

Foto von Engel62: <https://pixabay.com/de/daniel-d%C3%BCsentrieb-helferlein-123206/> (CC0 Public Domain Lizenz)

Was macht es dann schwierig?

Hello World vs. 50.000++ LOC

Disziplin (kleine Schritte, ...)

Aussagekräftige Testabdeckung

Legacy zu Clean Code?

Clean Code ist NICHT das Ziel

Hauptfokus: testbarer Code

Dann mal her mit den
X einfachen Schritten!



No Silver Bullet

Jedes Projekt individuell

Vorsicht beim Beheben von offensichtlichen Fehlern

Foto von stevepb: <https://pixabay.com/de/kugel-patrone-munition-kriminalit%C3%A4t-408636/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR | 46

A



Sicherheitsnetz
+ Tests

Foto von bella67: <https://pixabay.com/de/spinnennetz-mit-wasserperlen-netz-921039/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 47

B



Sanierung

Foto von KlausHausmann: <https://pixabay.com/de/bauarbeiter-bau-bauen-bohrhammer-921224/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 48

1

Golden Master



gegenwärtiges Verhalten
dokumentieren und erhalten

Foto von istara: <https://pixabay.com/de/gold-bar-goldbarren-reich-geld-296115/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 49

```
# suppose that our legacy code is this program called 'game'
$ game > GOLDEN_MASTER

# after some changes we can check to see if behaviour has changed
$ game > OUT-01
$ diff GOLDEN_MASTER OUT-01
# GOLDEN_MASTER and OUT-01 are the same

# after some other changes we check again and...
$ game > OUT-02
$ diff GOLDEN_MASTER OUT-02
# GOLDEN_MASTER and OUT-02 are different -> behaviour changed
```

Golden Master

(aka characterization tests)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 50

Konsolenausgaben abprüfen

```
@Before
public void init() {
    originalSysOut = System.out;
    consoleStream = new ByteArrayOutputStream();
    PrintStream printStream = new PrintStream(consoleStream);
    System.setOut(printStream);
}

@Test
public void testSimpleOutput() {
    System.out.println("Hallo Publikum!");
    System.out.print("Hallo Falk!");
    assertEquals("Hallo Publikum!\r\nHallo Falk!", consoleStream.toString());
}

@After
public void teardown() {
    System.setOut(originalSysOut);
}
```



1

Golden Master

Vorsicht bei Zufallsgeneratoren

```
1.9.3 > g = Random.new
1.9.3 > (1..10).map{g.rand(1000)}
=> [691, 362, 997, 692, 236, 532, 687, 616, 218, 702]
1.9.3 > g = Random.new
1.9.3 > (1..10).map{g.rand(1000)}
=> [865, 186, 89, 382, 894, 708, 769, 850, 452, 85]
1.9.3 > g = Random.new(1)
1.9.3 > (1..10).map{g.rand(1000)}
=> [37, 235, 908, 72, 767, 905, 715, 645, 847, 960]
1.9.3 > g = Random.new(1)
1.9.3 > (1..10).map{g.rand(1000)}
=> [37, 235, 908, 72, 767, 905, 715, 645, 847, 960]
```

Festlegen von Seeds (Pseudo-Random)

2

Subclass To Test

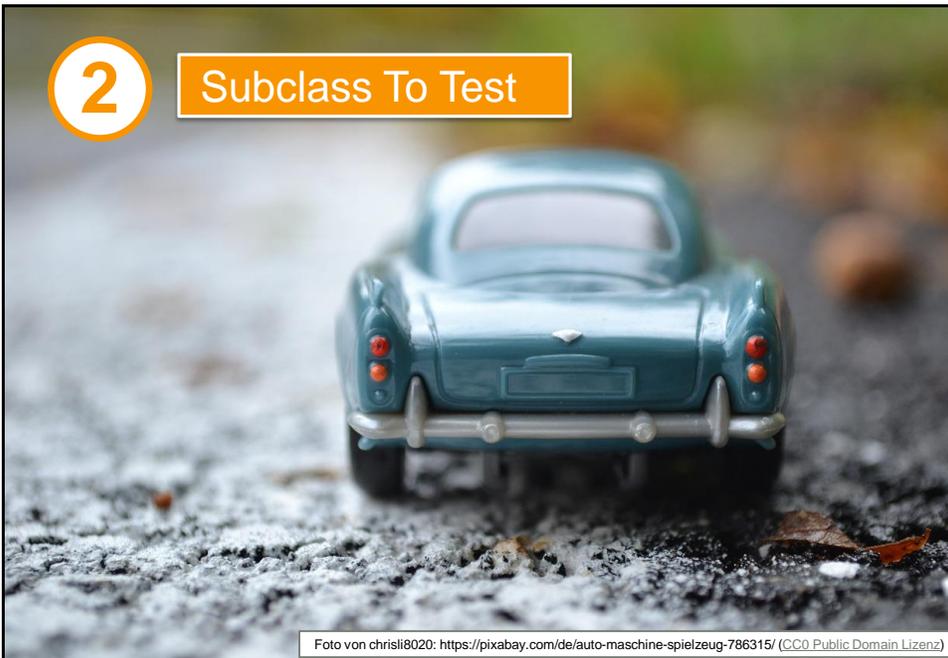


Foto von chrisli8020: <https://pixabay.com/de/auto-maschine-spielzeug-786315/> (CC0 Public Domain Lizenz)

2

Subclass To Test

Seam (Nahtstelle)

Ein Seam ist eine Stelle, an der man das Verhalten editieren kann, ohne direkt an dieser Stelle zu ändern.

Aufbrechen stark gekoppelter Abhängigkeiten

aka Extract and Override



3

Extract Pure Functions

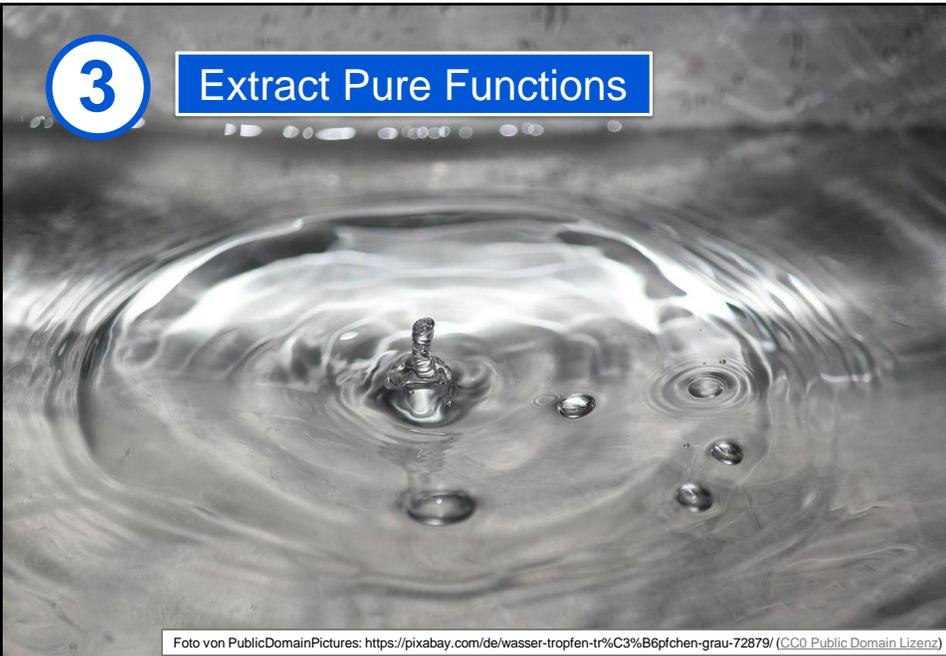


Foto von PublicDomainPictures: <https://pixabay.com/de/wasser-tropfen-tr%C3%B6pfchen-grau-72879/> (CC0 Public Domain Lizenz)

3

Extract Pure Functions



seiteneffektfrei

keine Statusänderung



"It's a classic,
we call it a Klassiker"

3

Extract Pure Functions

```
"pure function".substring(5);
```



```
UrlEncoder.encode("pure function");
```



```
Math.max(x, y);
```



```
System.out.println("unpure");
```



```
list.add(3);
```



```
Collections.sort(list);
```



3

Extract Pure Functions

Codestellen isolieren

Ziele

Separat testbar

Duplikation reduzieren

Live-Coding



Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 61

4 Remove Duplication



Don't Repeat Yourself

Beachte: Rule of Three

Foto von AdinaVoicu: <https://pixabay.com/de/zwilling-schwestern-liebe-m%C3%A4dchen-948713/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 62

Live-Coding

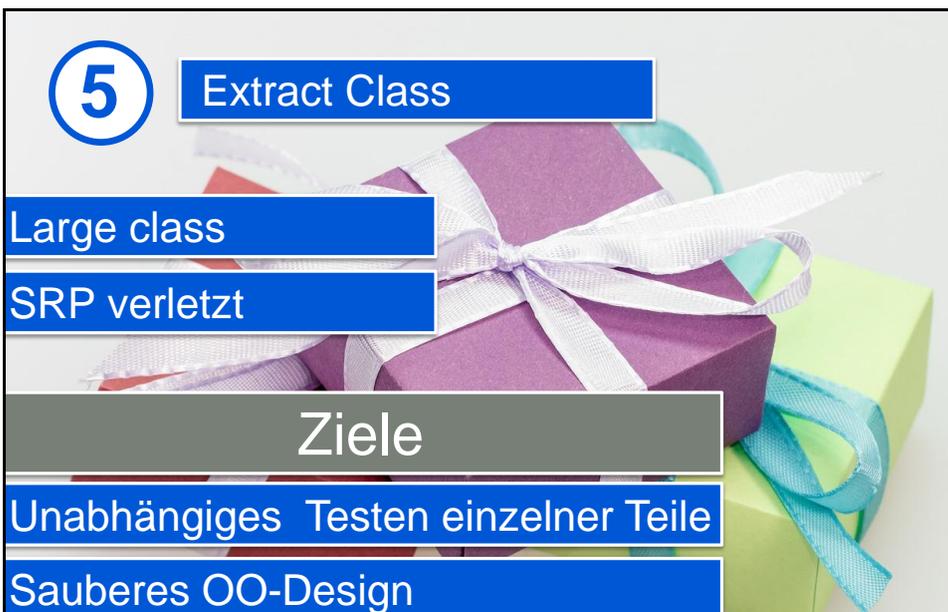


Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 63

5 Extract Class



Large class

SRP verletzt

Ziele

Unabhängiges Testen einzelner Teile

Sauberes OO-Design

Foto von blickpixel: <https://pixabay.com/de/weihnachtsdekoration-pakete-geschenk-570797/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 64

Live-Coding



Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 65

~~Hätten wir mehr Zeit ...~~

- 6 Dependency Inversion
- 7 Test non-public member
- 8 Mocking Framework

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 66

6

Dependency Inversion

Entkoppeln
durch **explizites**
Setzen der
Abhängigkeiten

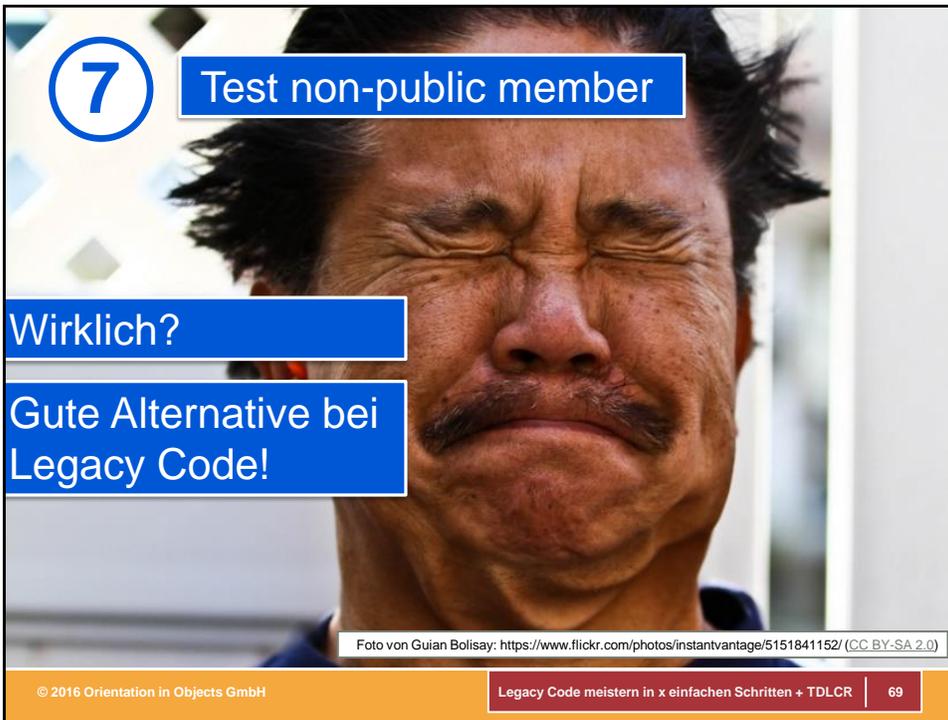


Foto von tatin: <https://pixabay.com/de/zigarette-rauch-tabak-106610/> (CC0_Public Domain Lizenz)

Live-Coding



Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0_Public Domain Lizenz)



7 Test non-public member

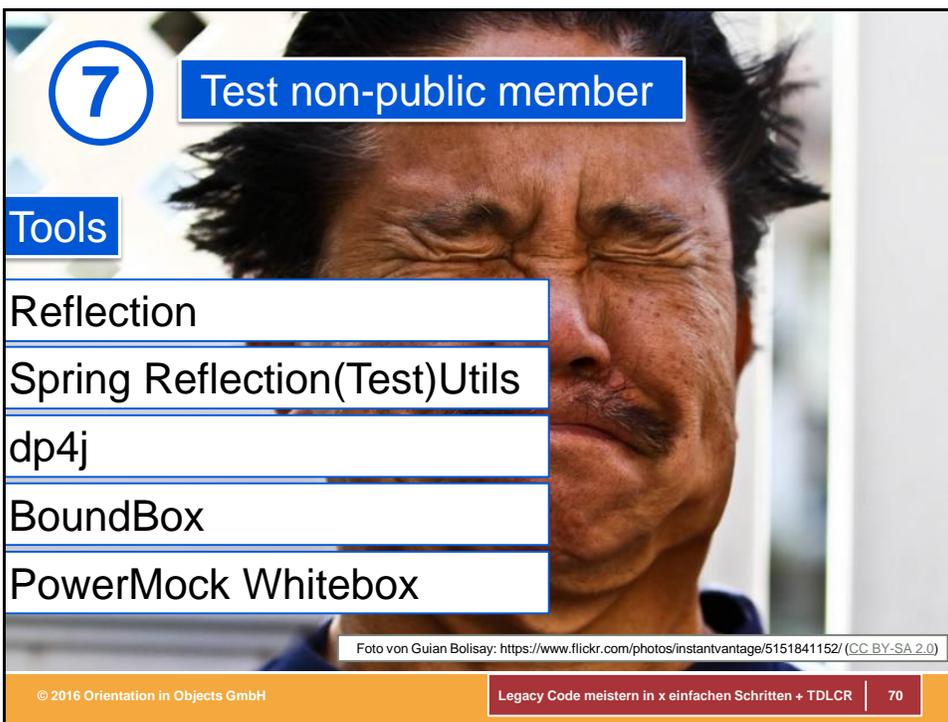
Wirklich?

Gute Alternative bei Legacy Code!

Foto von Guian Bolisay: <https://www.flickr.com/photos/instantvantage/5151841152/> (CC BY-SA 2.0)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 69



7 Test non-public member

Tools

- Reflection
- Spring Reflection(Test)Utils
- dp4j
- BoundingBox
- PowerMock Whitebox

Foto von Guian Bolisay: <https://www.flickr.com/photos/instantvantage/5151841152/> (CC BY-SA 2.0)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 70

Live-Coding

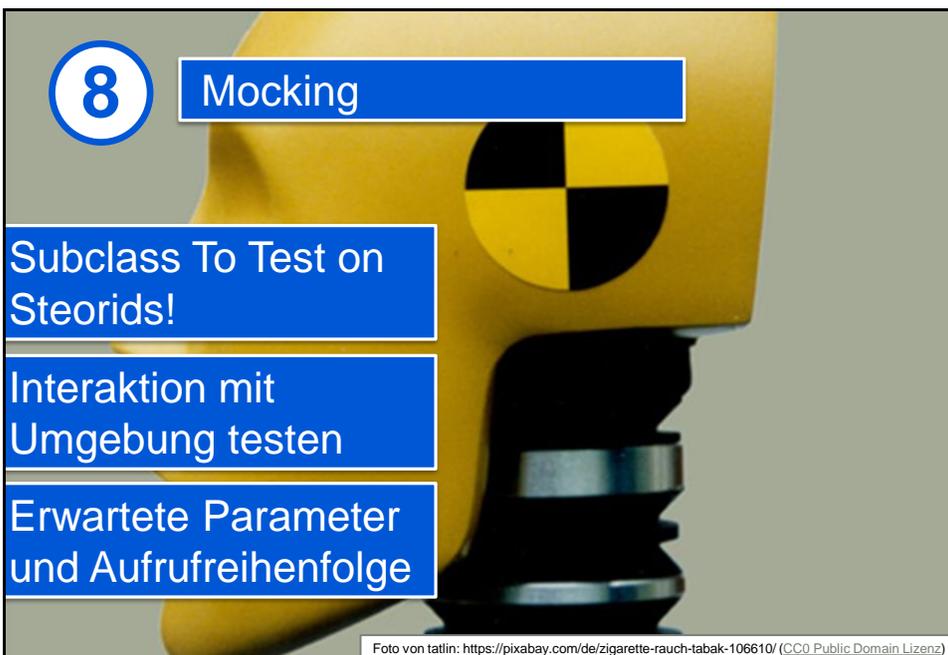


Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 71

8 Mocking



Subclass To Test on Steroids!

Interaktion mit Umgebung testen

Erwartete Parameter und Aufrufreihenfolge

Foto von tatlin: <https://pixabay.com/de/zigarette-rauch-tabak-106610/> (CC0_Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 72

Live-Coding



Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0 Public Domain Lizenz)

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 73



Orientation in Objects

Zusammenfassung

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR 74

| | | | |
|---|------------------------|---|--|
| ① | Golden Master |  |  Orientation in Objects |
| ② | Subclass To Test |  | |
| ③ | Extract Pure Functions |  | |
| ④ | Remove Duplication |  | |
| ⑤ | Extract Class |  | |
| ⑥ | Dependency Inversion |  | |
| ⑦ | Test non-public member |  | |
| ⑧ | Mocking Framework |  | |

© 2016 Orientation in Objects GmbH | Legacy Code meistern in x einfachen Schritten + TDLCR | 75

Sicherheitsnetz



| | | |
|---|---------------|---|
| ① | Golden Master |  |
|---|---------------|---|

© 2016 Orientation in Objects GmbH | Legacy Code meistern in x einfachen Schritten + TDLCR | 76

Was mache ich bei GUI-Anwendungen?



```

3 void account(int minuten, int stunde, int minute) {
  System.out.println(String.format("Berechne Gespräch mit
boolean mondschein = false;
double preis = 0;
this.
// Mon
= gebuehr: double - Kunde
if (st
= tarif: Tarif - Kunde
= account(int minuten, int stunde, int minute): void - Kunde
// Geb
= berechnePreis(int minuten, double d): double - Kunde

```



```

1 public static void main(String... args) throws Exception {
  WebDriver driver = new FirefoxDriver();
  driver.get("http://www.retest.de");
  while (true) {
    List<WebElement> links = driver.findElements(By.tagName("a"));
    links.get(random.nextInt(links.size())).click();
    Thread.sleep(500);
    List<WebElement> fields =
      driver.findElements(By.xpath("//input[@type='text']"));
    WebElement field = fields.get(random.nextInt(fields.size()));
    field.sendKeys(randomString());
    Thread.sleep(500);
  }
}

```



https://entwicklertag.de/frankfurt/2016/sites/entwicklertag.de/frankfurt.2016/files/slides/Bei%20Uns%20testen%20lauter%20Affen_0.pdf

Leichtere Testbarkeit



- 2 Subclass To Test
- 3 Extract Pure Functions
- 5 Extract Class
- 6 Dependency Inversion
- 7 Test non-public member
- 8 Mocking Framework



Besseres Verständnis



③ Extract Pure Functions



④ Remove Duplication



⑤ Extract Class



T
D
L
C
R



Neue
Features
ausliefern ...

Aufhören Legacy Code zu schreiben!

- sonst wird die Codebasis nur schlimmer und man entfernt sich immer mehr davon, jemals Tests hinzuzufügen
- **Keine neuen Features mehr ohne Unit-Testing!**
- Sprout Method + Wrap Method als Hilfen

Sprout Method

```
public int berechne(int a, int b) {  
    int c = a + b;  
    // weitere wichtige Aufgaben  
  
    // neues Verhalten  
    c = verdoppeleResult(c);  
  
    // weitere wichtige Aufgaben  
    return c;  
}
```

Aufruf aus
Bestandscode

neue Methode,
testgetrieben
entwickeln

```
protected int verdoppeleResult(int result) {...}
```

Sprout Method und TDD (1)



```
// nicht kompilierenden Test schreiben

@Test
public void testVerdoppeleResult() {
    assertEquals(2, rechner.verdoppeleResult(1);
}

// Kompilerfehler beheben

protected int verdoppeleResult(int result) {
    return 0;
}

// Test ist rot
```

Sprout Method und TDD (2)



```
// einfachste Lösung, damit Test grün wird

protected int verdoppeleResult(int result) {
    return 2;
}

// nichts zu refactoren, von vorn beginnen

@Test
public void testVerdoppeleResult() {
    assertEquals(2, rechner.verdoppeleResult(1);
    assertEquals(4, rechner.verdoppeleResult(2);
}

// Test wieder rot, Lösung generalisieren
```

Wrap Method

- ähnlich zu Sprout Method, neues Verhalten aber vor oder am Ende der zu ändernden Methode

```
public int berechne(int a, int b) {  
    logResult(c);  
    return berechnePrivate(a, b);  
}  
  
private int berechnePrivate(int a, int b) {  
    int c = a + b;  
    // weitere wichtige Aufgaben  
    return c;  
}
```

```
protected void logResult(int result) {...}
```

Live-Coding

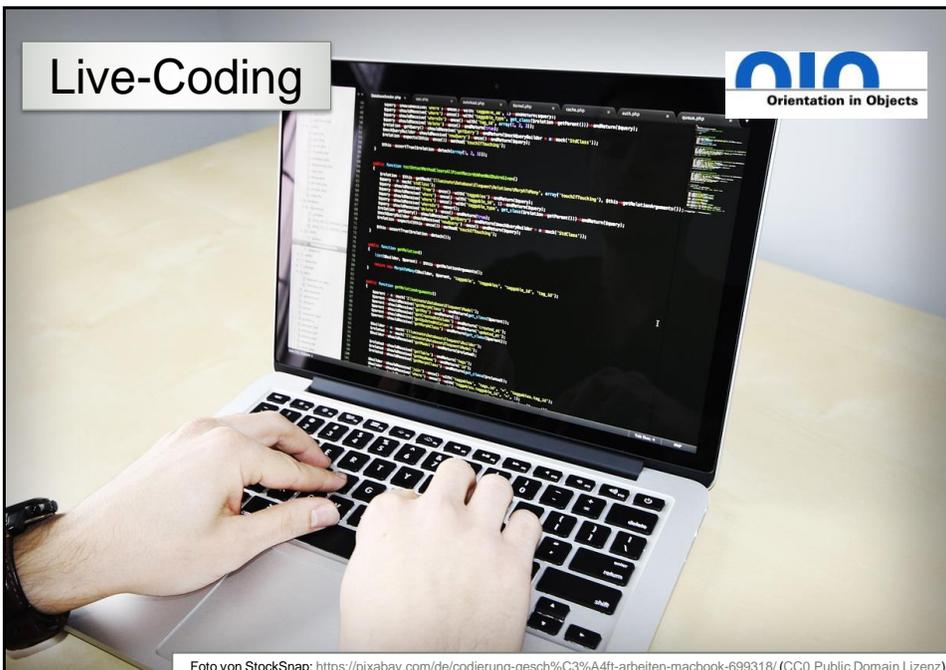


Foto von StockSnap: <https://pixabay.com/de/codierung-gesch%C3%A4ft-arbeiten-macbook-699318/> (CC0 Public Domain Lizenz)

Zusammenfassung: Sprout Method und TDD

- Test schreiben vor der Integration der neuen Funktionalität
 - Red – Green – Refactor
- alte Funktionalität wird gleich mitgetestet
 - Testbett wird bei jeder Erweiterung besser
- uninteressante Abhängigkeiten durch Mocks ersetzen

Zusammenfassung (2. Teil)

- Sicherheitsnetz spannen: Golden Master
- Legacy Code refactoren => Ziel testbarer Code
 - Dependencies aufbrechen
 - Codestellen separieren
- Tests immer hinzufügen bei
 - Bugfixing
 - Erweitern
- TDD konsequent beim Schreiben von neuem Code und auch bei der Aufruf-Integration von neuem Code
- nach und nach alten Code ebenfalls mit Unit-Tests versehen

Tools

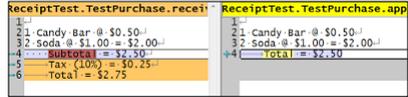
Eclipse Metrics



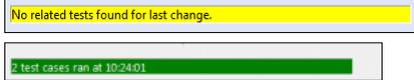
MoreUnit



Approval Tests



Infinittest



Code Coverage

| | | | |
|-------------------------------|---|-----------------------|---------|
| ▷ | U | Tarif.java | 100,0 % |
| src/test/java | | | 91,6 % |
| de.oio.refactoring.badtelefon | | | 91,6 % |
| ▷ | U | TarifeRunnerTest.java | 85,2 % |
| ▷ | U | TarifeRunnerTest | 81,4 % |
| ▷ | U | KundeTests.java | 100,0 % |

© 2016 Orientation in Objects GmbH

Legacy Code meistern in x einfachen Schritten + TDLCR

89

Legacy Code Retreat



<https://github.com/jbrains/trivia>

Foto von Jmabel: https://commons.wikimedia.org/wiki/File:Seattle_-_Budokan_Dojo_judo_demo_04.jpg?uselang=de (CC BY-SA 3.0 Lizenz)

© 2016 Orientation in Objects GmbH

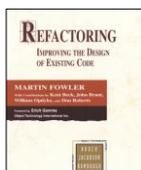
Legacy Code meistern in x einfachen Schritten + TDLCR

90

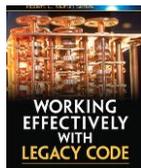
Links

- Code-Beispiel der Live-Demo (**Legacy Code meistern in x ...**)
 - <https://github.com/sippsack/BadTelefon-Refactoring-Legacy-Code>
- Code-Beispiel der Live-Demo (**TDLCR**)
 - <https://github.com/sippsack/BadTelefon-Test-Driven-Legacy-Code-Refactoring>
- anderes Code-Beispiel für Legacy Code
 - <https://github.com/jbrains/trivia>
- Blog: Techniken zu Legacy Code-Retreat
 - <http://blog.adrianbolboaca.ro/2014/04/legacy-coderetreat/>

Literaturhinweise



- Refactoring
 - **Sprache: Englisch**
 - **Gebunden - 464 Seiten - Addison Wesley**
 - **Erscheinungsdatum: 1. Juni 1999**
 - **ISBN: 0201485672**



- Working Effectively with Legacy Code
 - **Sprache: Englisch**
 - **Gebunden**



Fragen ?

Orientation in Objects GmbH
Weinheimer Str. 68
68309 Mannheim
www.oio.de
info@oio.de



**Vielen Dank für ihre
Aufmerksamkeit !**

Orientation in Objects GmbH
Weinheimer Str. 68
68309 Mannheim
www.oio.de
info@oio.de