



Refactoring mit der Mikado-Methode

FALK SIPPACH

Developer Week 2021

Mittwoch, 30.06.2021

Refactoring mit der Mikado-Methode



Zusammenfassung

Viele von uns haben tagtäglich mit Legacy-Code zu tun. Mal eben schnell etwas umzubauen, scheitert typischerweise an den fehlenden Tests, zudem ist der Quellcode oft überhaupt schlecht testbar.

In diesem Vortrag wird anhand von praktischen Codebeispielen gezeigt, wie man zunächst ein automatisiertes Sicherheitsnetz aufspannt. Anschließend werden komplexere Refactorings durchgeführt, ohne jedoch zu viele Baustellen gleichzeitig aufzureißen. Die Mikado-Methode hilft dabei, den Überblick zu behalten und in möglichst kleinen und nachvollziehbaren Schritten vorzugehen. Das Ziel ist das Aufbrechen stark gekoppelter Abhängigkeiten, um so neue Tests hinzufügen zu können. Zudem wird der Code besser lesbar sein und lässt sich so auch leichter warten und wiederverwenden.

Falk Sippach

- Softwarearchitekt, Berater, Trainer bei embarc
- früher bei Orientation in Objects (OIO), Trivadis

Schwerpunkte:

- Architekturberatung und -bewertung
- Cloud- und Java-Technologien



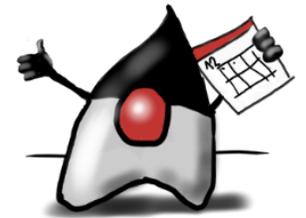
fs@embarc.de



@sipp sack



→ [xing.to/fsi](https://www.xing.to/fsi)





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!



9



31



Michael Simons @rotnroll666 · 8. Nov.

@GeertjanW like a minibarcamp? 🍺



2



Agenda



- 1** Einstieg
- 2** Legacy Code
- 3** Mikado Methode
- 4** Reale Legacy Projekte
- 5** Ausblick und weitere Informationen



Agenda



- 1 Einstieg**
- 2 Legacy Code
- 3 Mikado Methode
- 4 Reale Legacy Projekte
- 5 Ausblick und weitere Informationen

1

The **Mikado Method** is a **structured way** to make **significant changes** to **complex code**.

... for **performing changes** to a **system that's too large for analyze-then-edit**, which means basically **any production system** in the world.

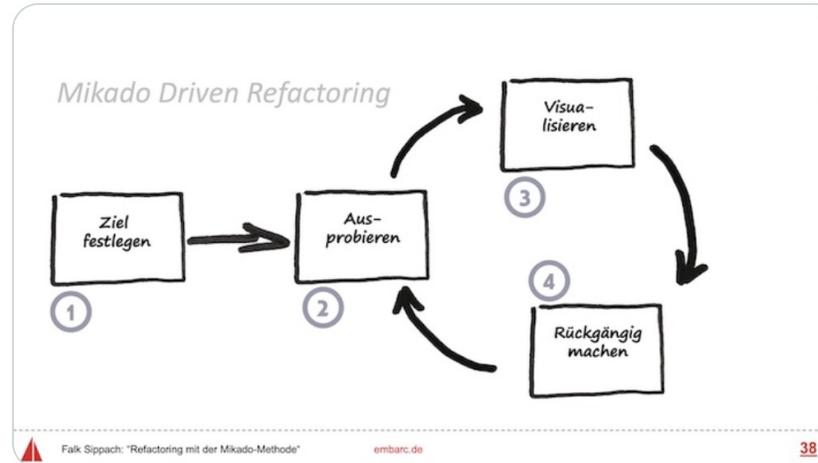
(Ola Ellnestam, Daniel Brolund: "The Mikado Method")





Falk Sippach @sipsack · 7 Std.

Heute um 13 Uhr darf ich bei den #ittage 365 einen Vortrag zu Refactoring mit der Mikado-Methode. Hier sind schon mal die Folien: embarc.de/mikado-ittage-... @InformatikAktue @embarced



1



3



4



Dierk König
@mittie

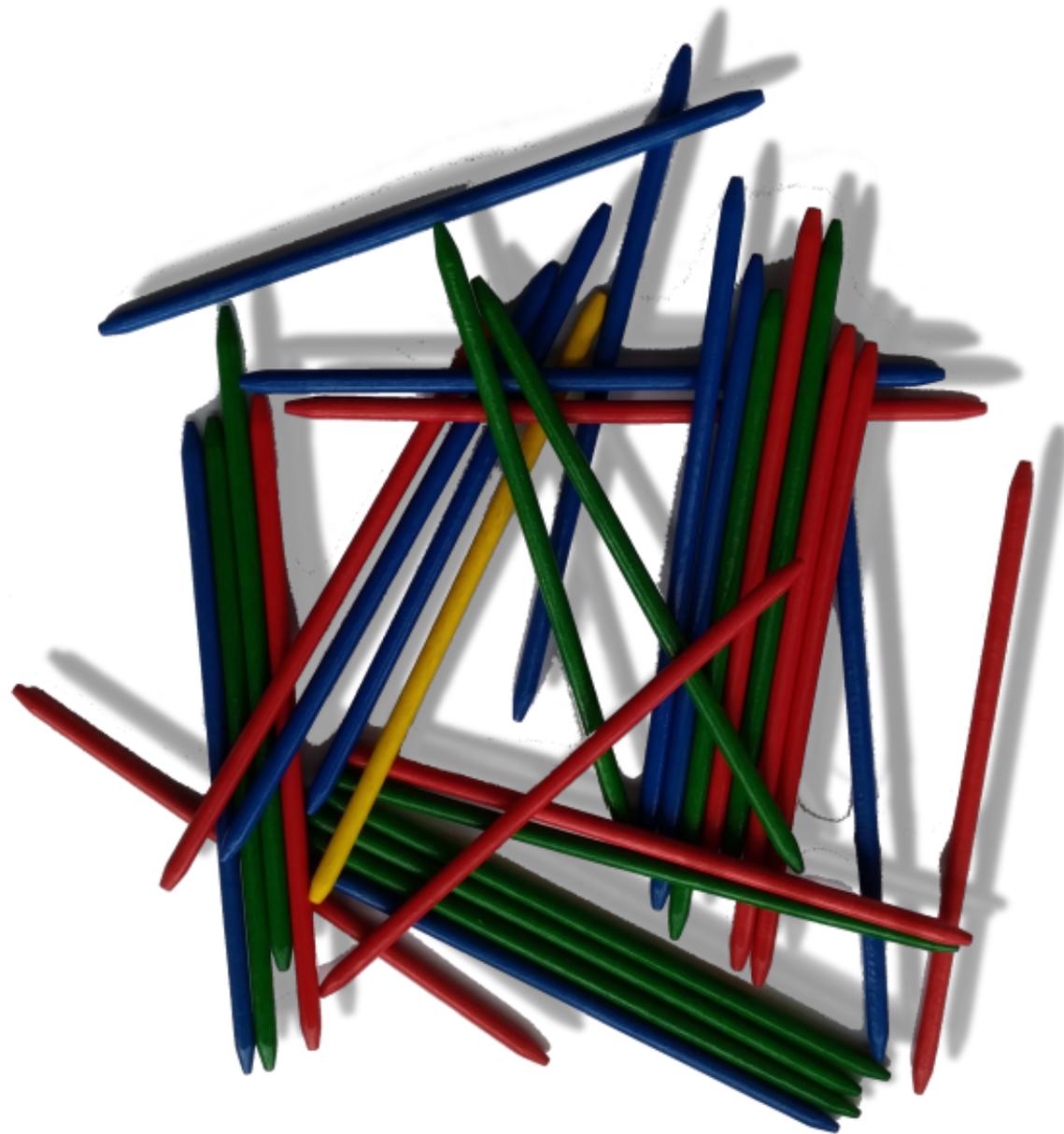
Antwort an @sipsack @RalfDMueller und 2 weitere Personen

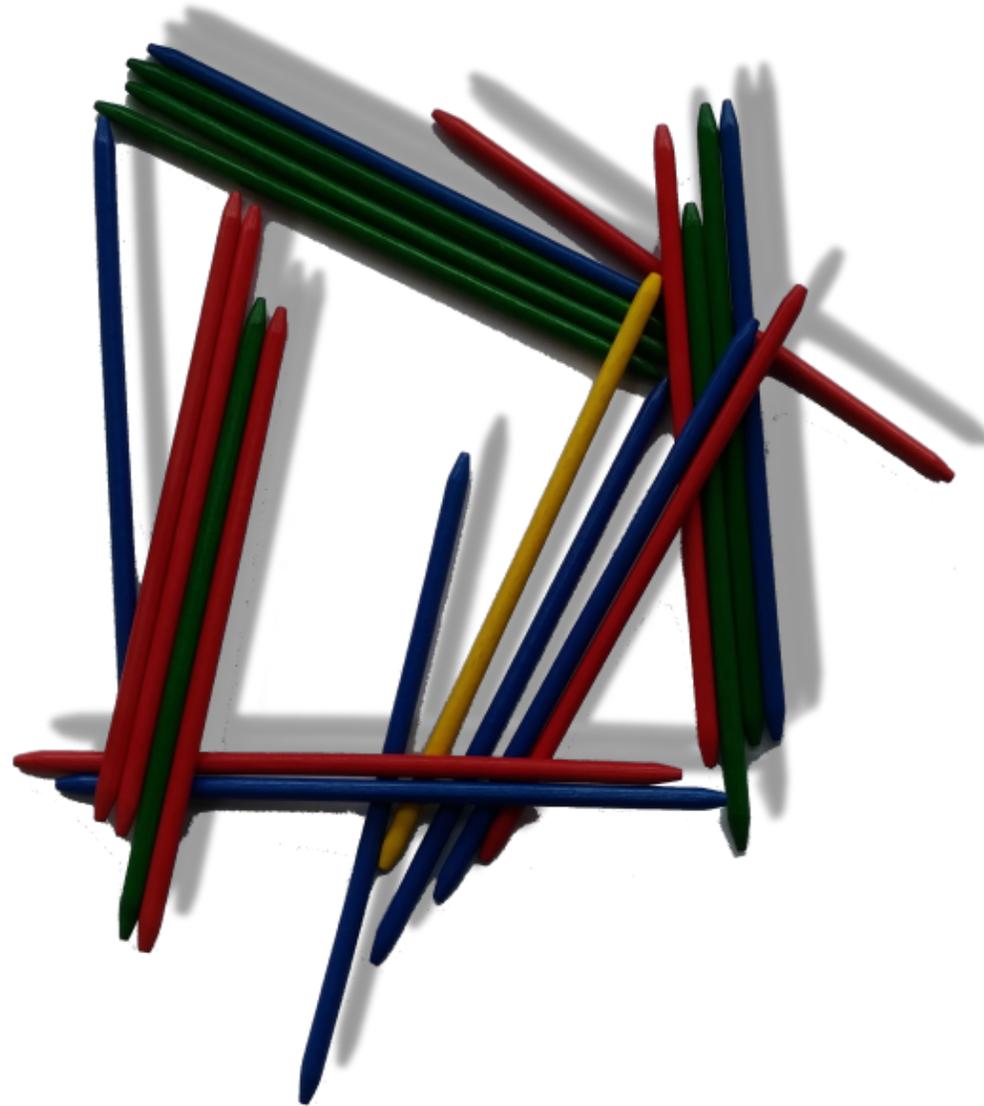
besser als Refactorings mit Domino Methode...

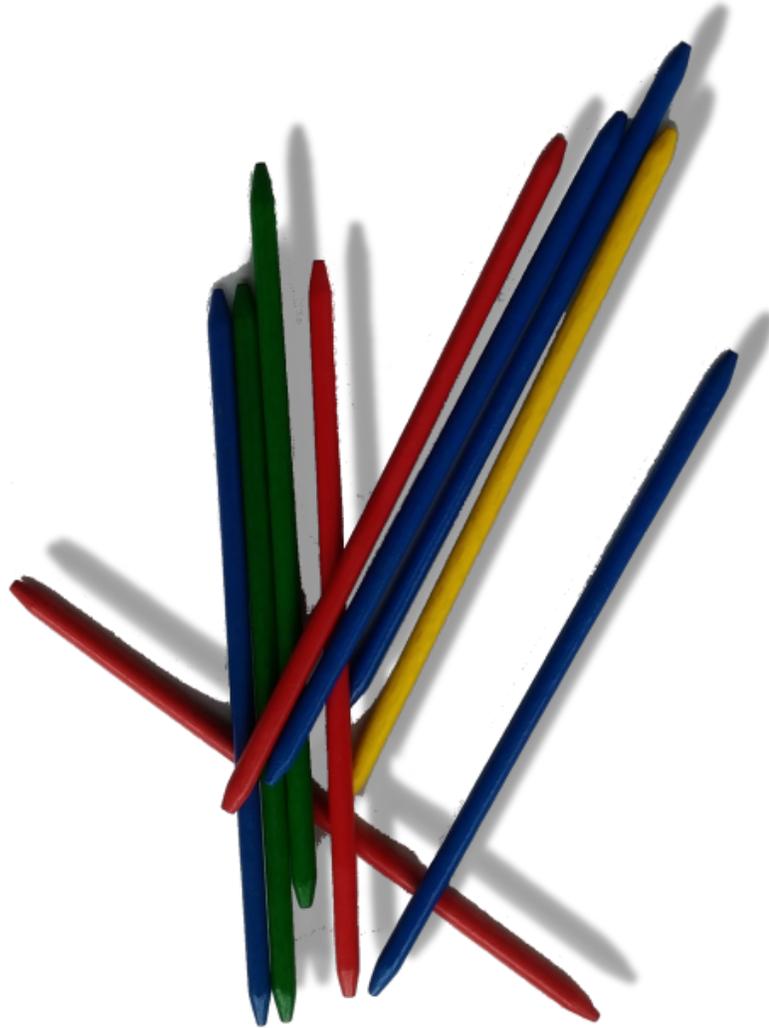
[Tweet übersetzen](#)

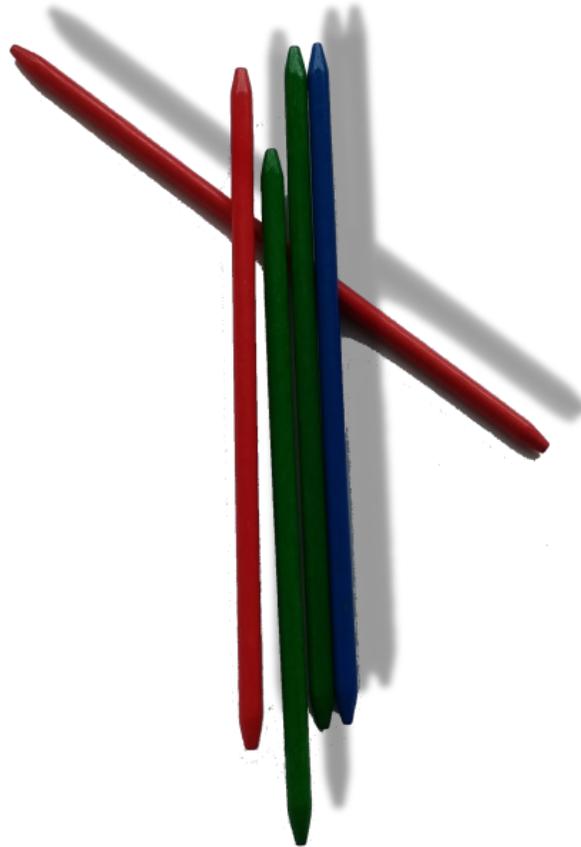
1:28 nachm. · 17. Juni 2021 · Twitter for iPhone

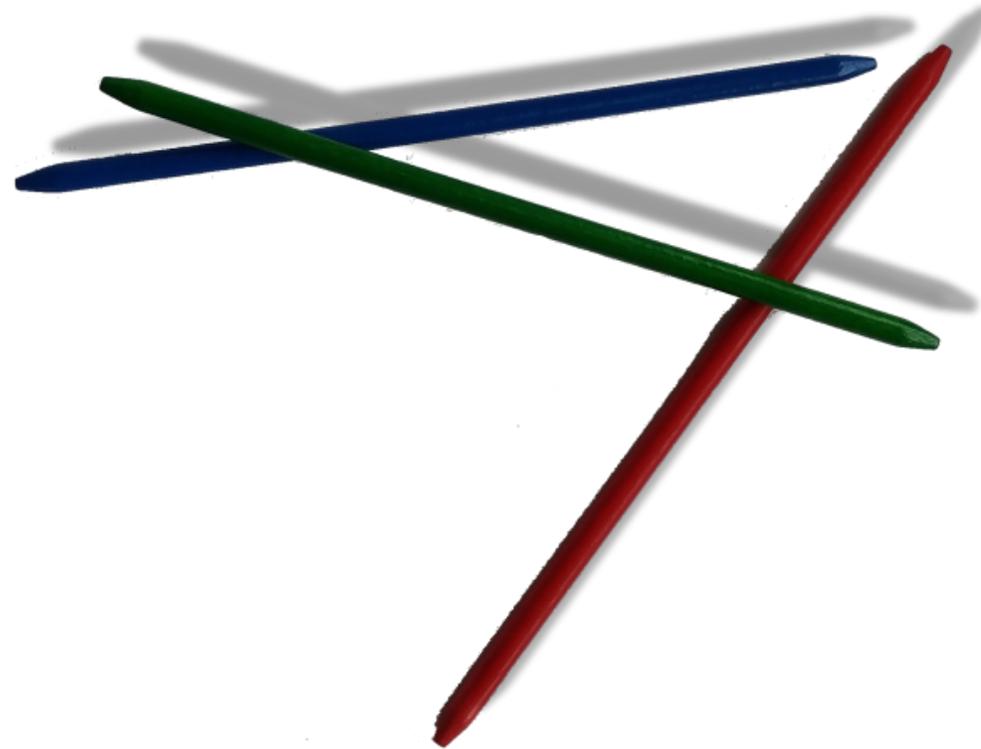












Agenda



- 1 Einstieg
- 2 Legacy Code**
- 3 Mikado Methode
- 4 Reale Legacy Projekte
- 5 Ausblick und weitere Informationen

2

Code ändern? Nur wie?

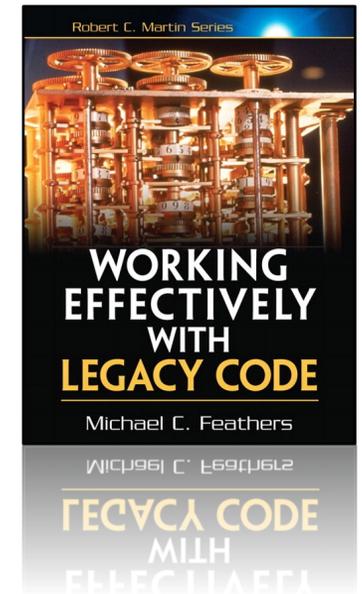
***Brownfield-Projekt. Code von anderen.
Fehlende Dokumentation, kaum Tests.
Änderungen geraten außer Kontrolle.***





Michael Feathers

“ Code
without tests
is bad code.”





J. B. Rainsberger

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



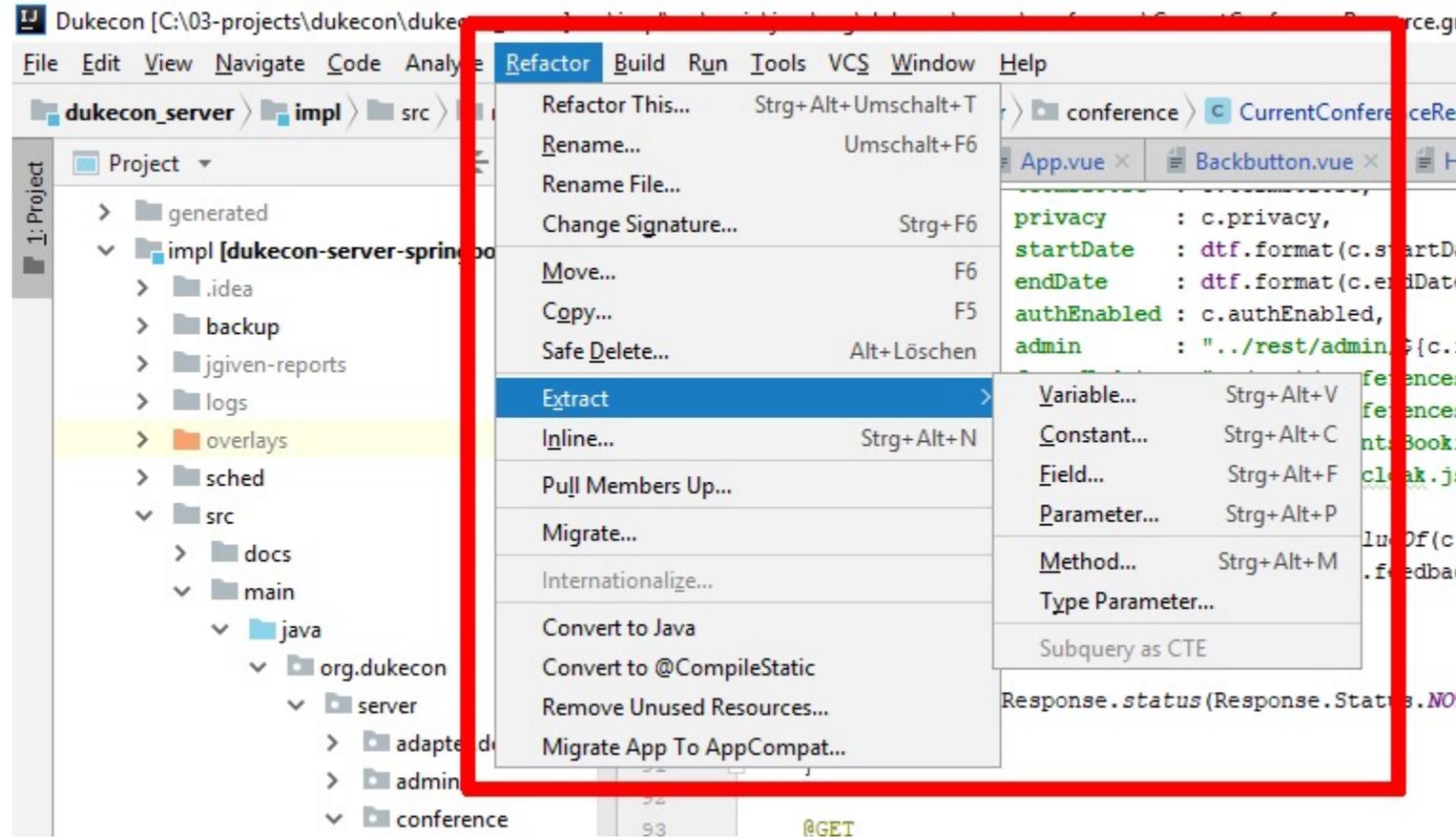
Foto von PublicDomainPictures, [CC0 Public Domain Lizenz](https://pixabay.com/de/menschen-abdeckung-schrei-314481/), <https://pixabay.com/de/menschen-abdeckung-schrei-314481/>

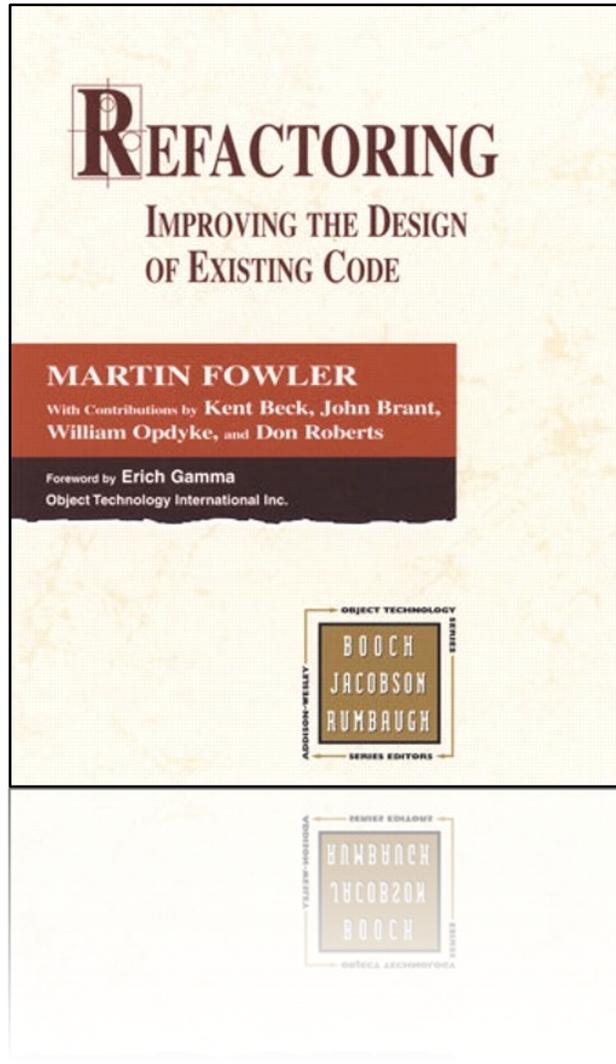
Gründe für Refactoring

Verstehen
Fehler beheben
Neues Feature
Optimierung

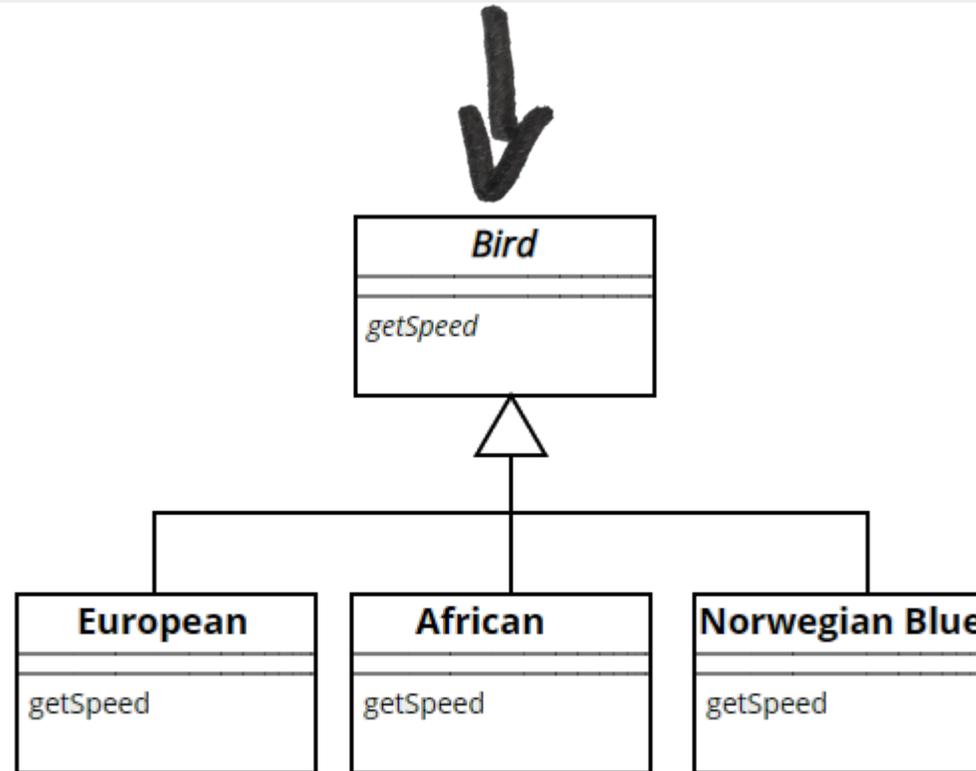


Toolunterstützung bei einfachen Refactorings





```
double getSpeed() {  
    switch (_type) {  
        case EUROPEAN:  
            return getBaseSpeed();  
        case AFRICAN:  
            return getBaseSpeed() - getLoadFactor() * _numberOfCoconuts;  
        case NORWEGIAN_BLUE:  
            return (_isNailed) ? 0 : getBaseSpeed(_voltage);  
        }  
        throw new RuntimeException ("Should be unreachable");  
    }  
}
```



Agenda



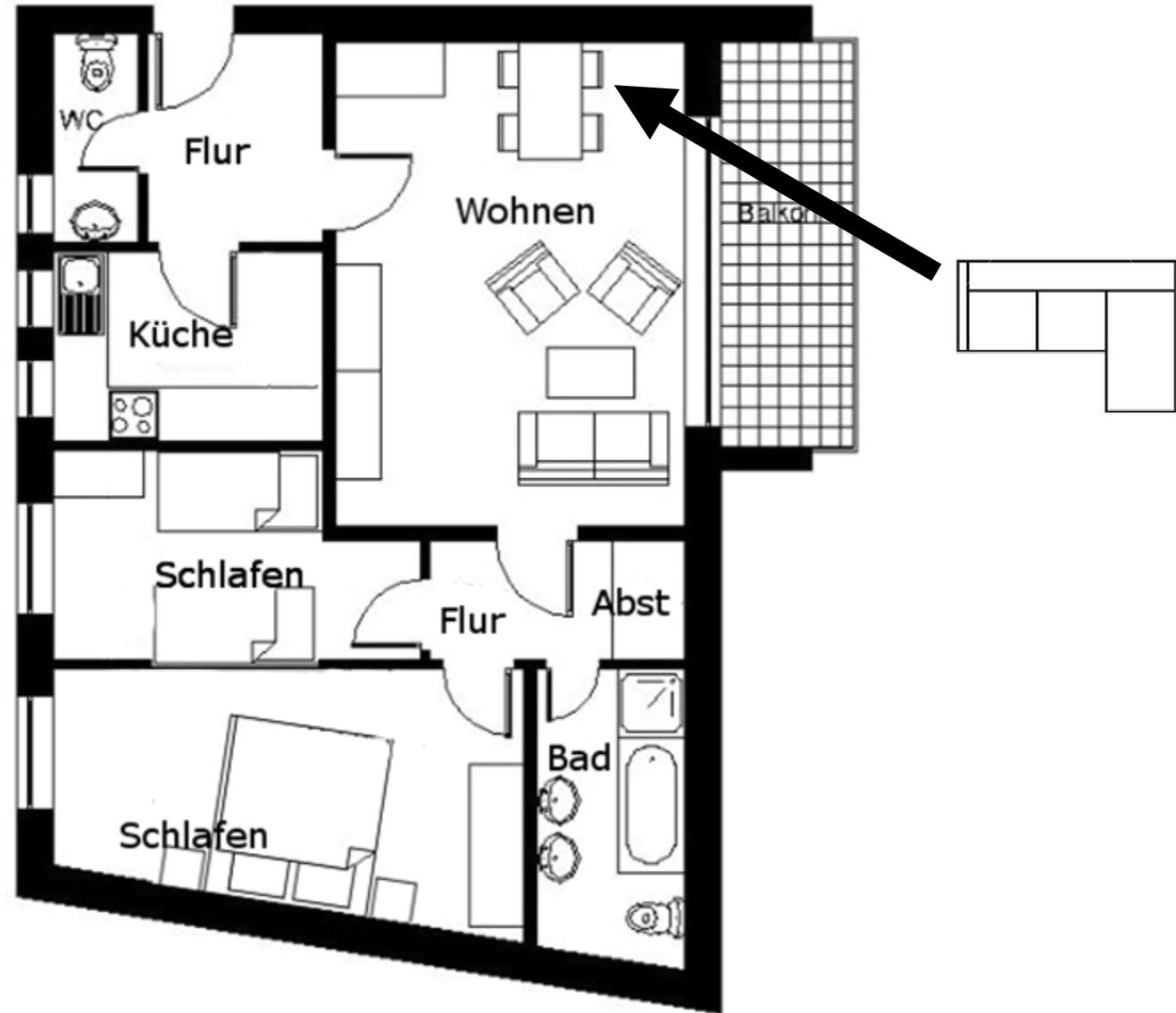
- 1 Einstieg
- 2 Legacy Code
- 3 Mikado Methode**
- 4 Reale Legacy Projekte
- 5 Ausblick und weitere Informationen

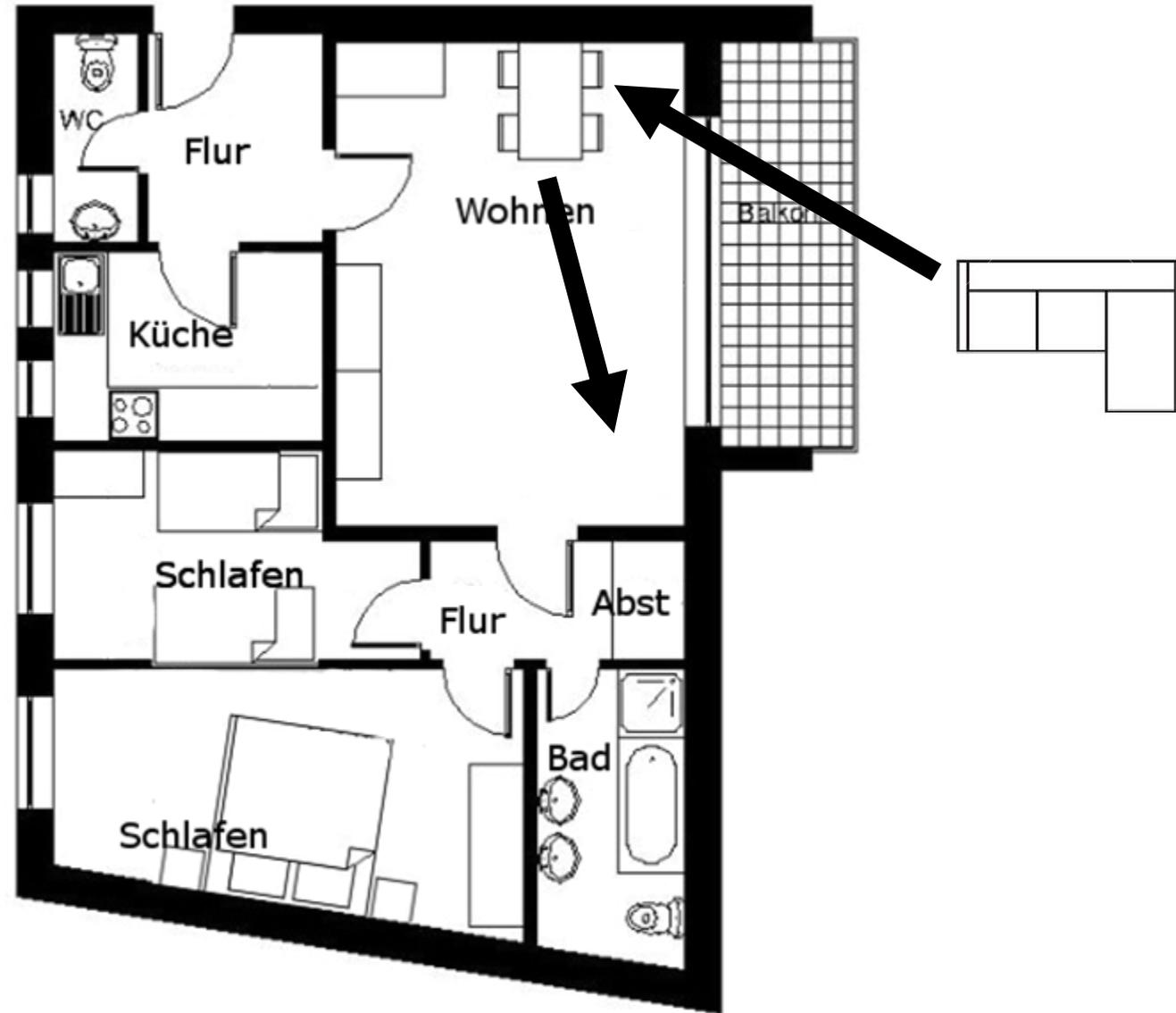
3

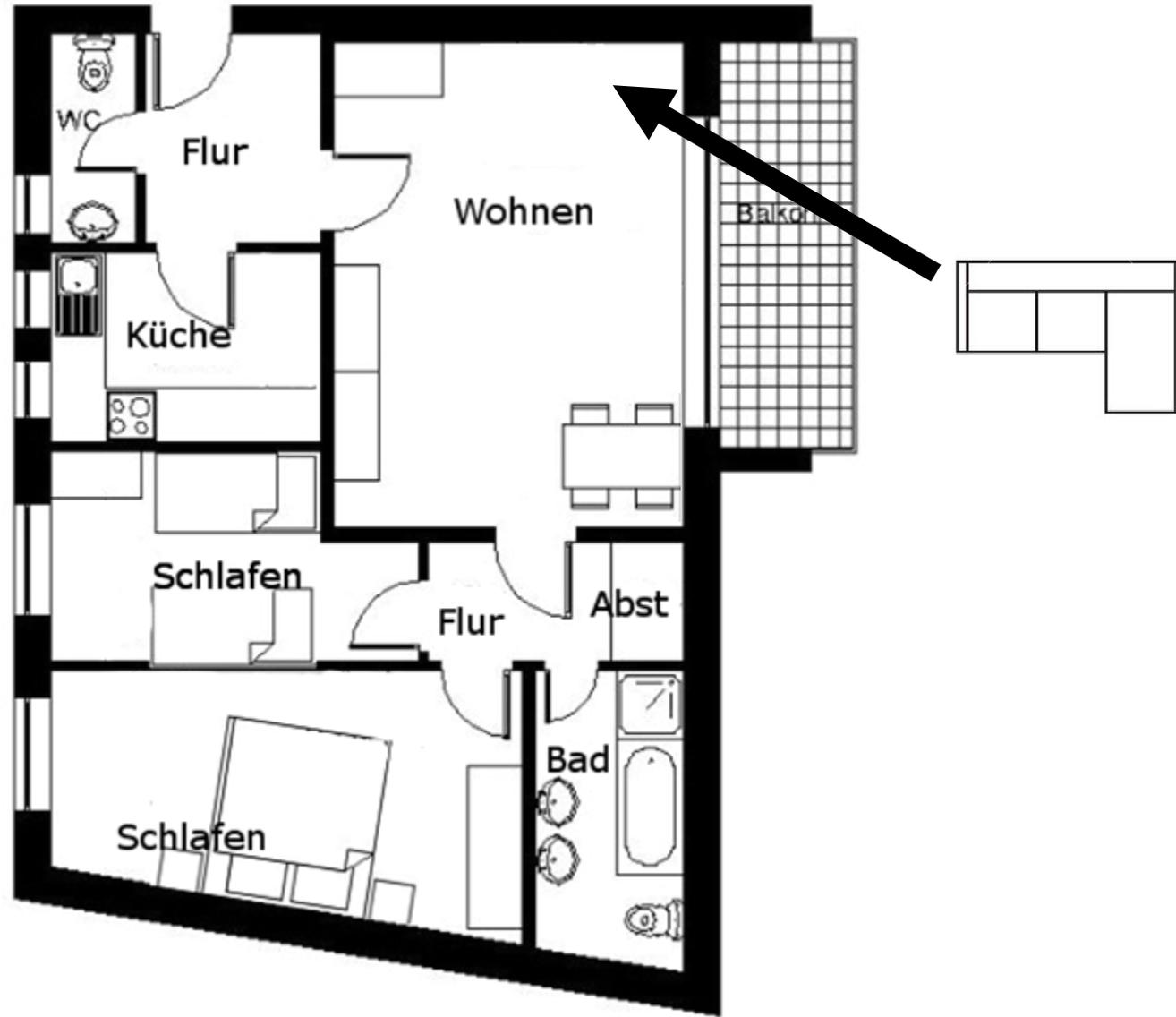
Mikado Methode für Dummies

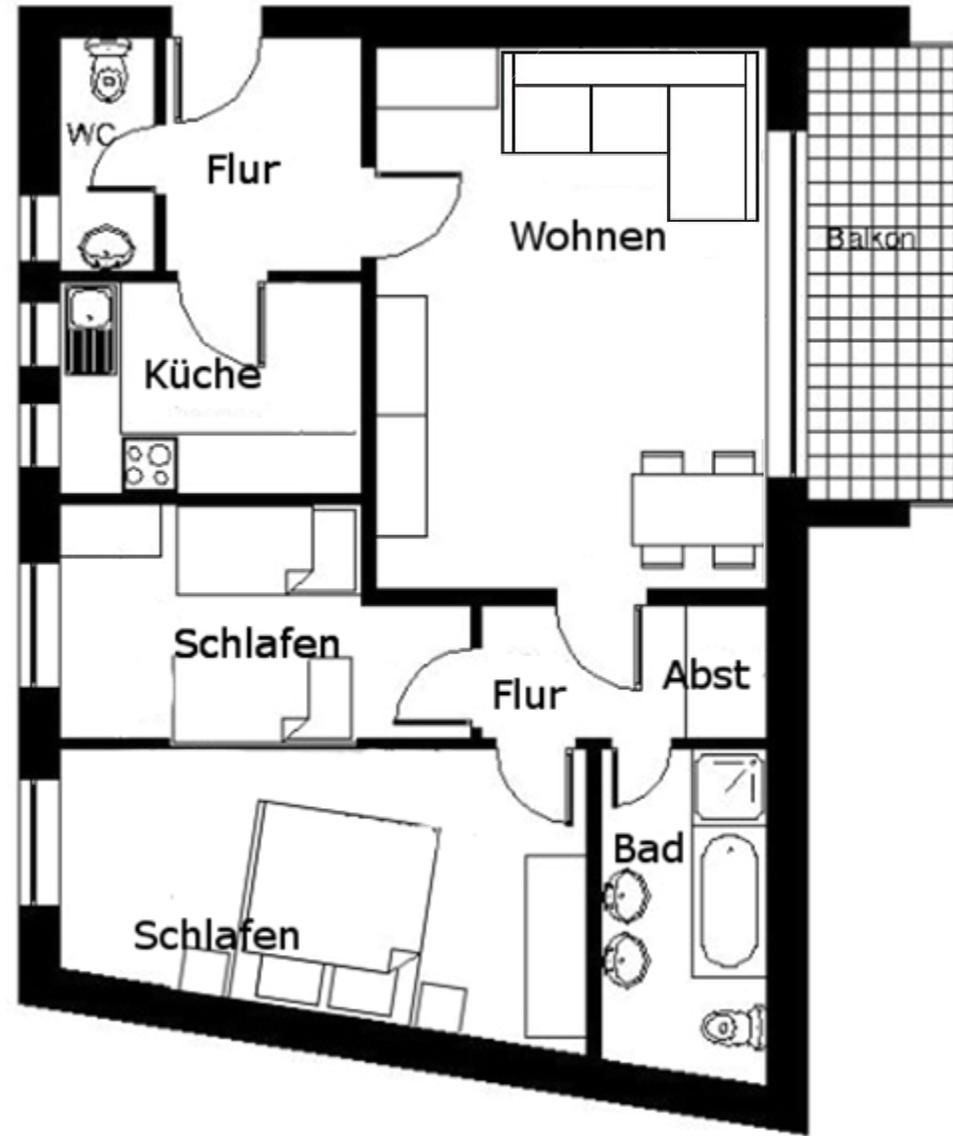
Möbel rücken









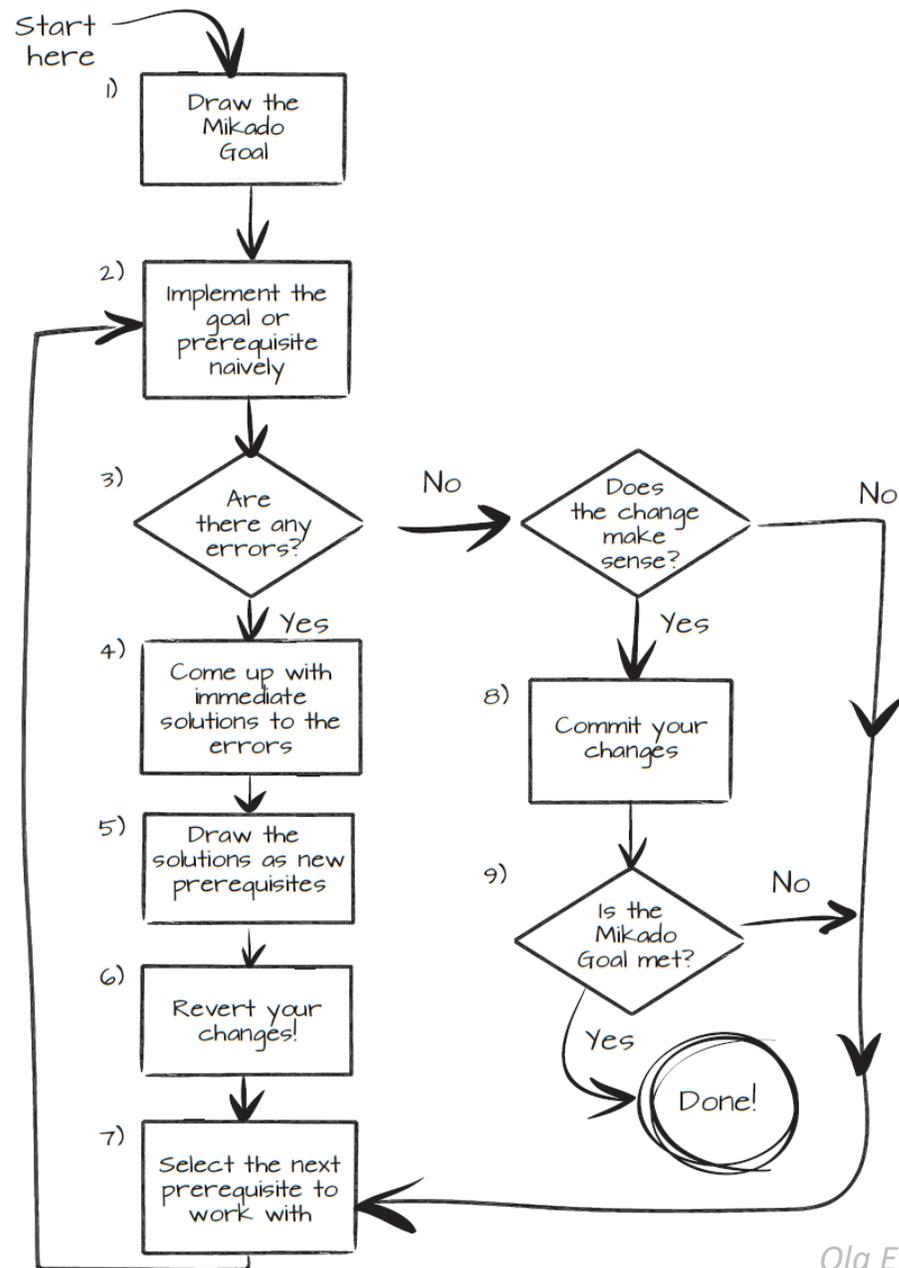


The Mikado Method can help you **visualize, plan, and perform** business value–focused **improvements** over several iterations and increments of work, **without** ever having **a broken codebase** during the process.

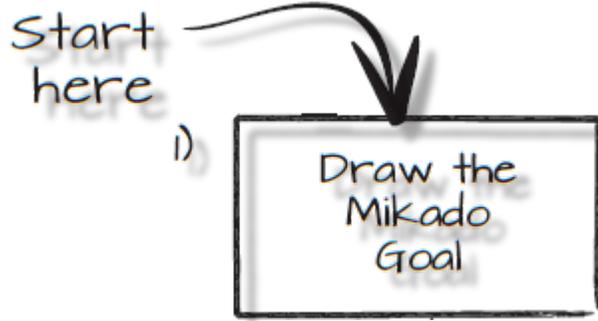
(Ola Ellnestam, Daniel Brolund: "The Mikado Method")



Ablauf im Großen:

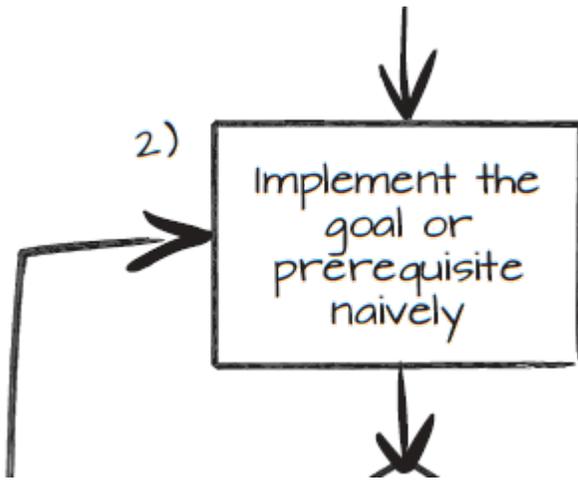


Ola Ellnestam, Daniel Brolund: "The Mikado Method"

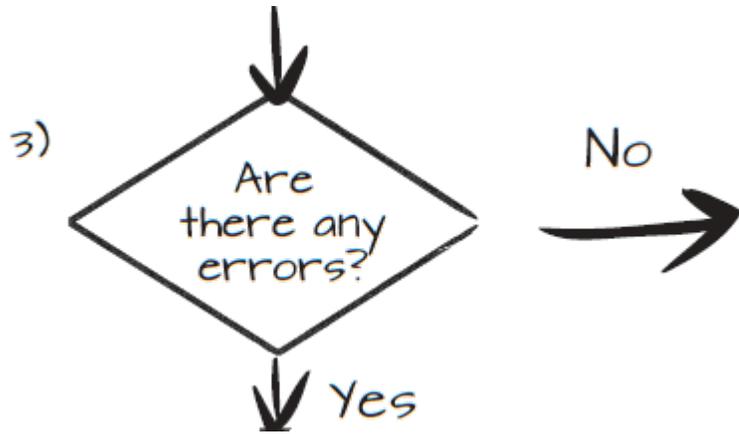


***Konkrete Aufgabe, z. B. User Story.
Auf Papier schreiben.***

**Mondscheinzeit-
berechnungslogik
verbessern**



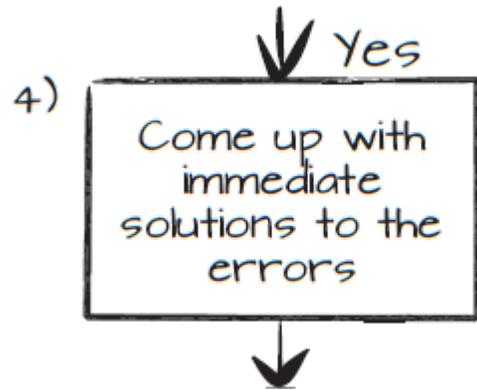
***Direkt eine einfache Lösung ausprobieren.
Experimentieren statt analysieren.
Hilfe durch Compiler und Tests.***



Auftretende Fehler haben die Ursache in weiteren Abhängigkeiten.

Zu Schritt 8, wenn keine Fehler.

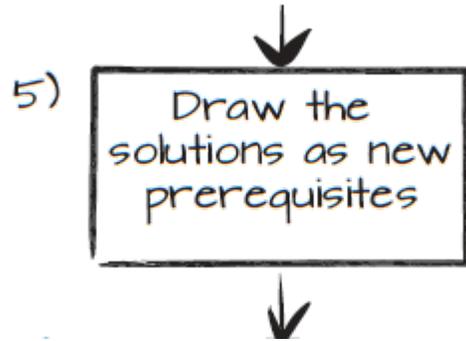
Vorsicht bei Laufzeitfehlern (Nullpointer).



Direkt sofortige Lösungen zu den Fehlern finden.

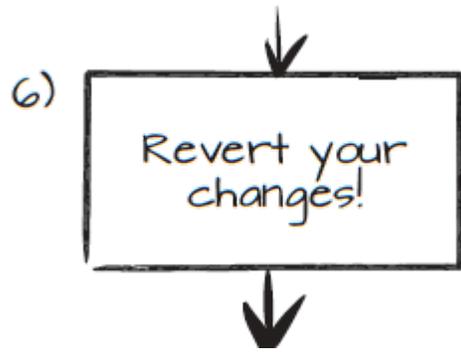
Überanalysieren vermeiden.

Weiter zugrundeliegende Einschränkungen werden in späteren Iterationen behandelt.



***Lösung aufzeichnen und mit Vorgänger verbinden.
Wissen über das System aufbauen.***

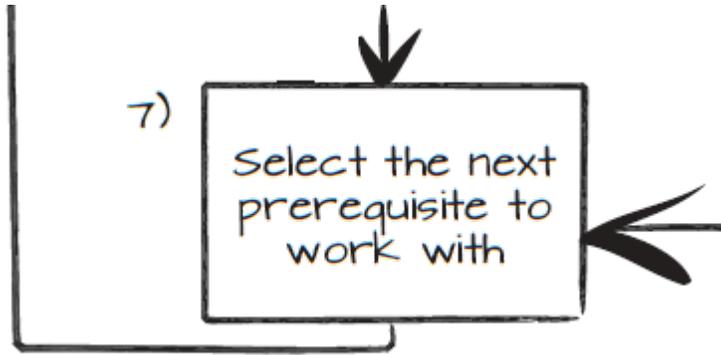




Bei Fehlern immer zurückrollen.

Weitere Bearbeitung auf kaputten Zustand sehr fehleranfällig.

Zurückgerollte Informationen stecken im Graph.

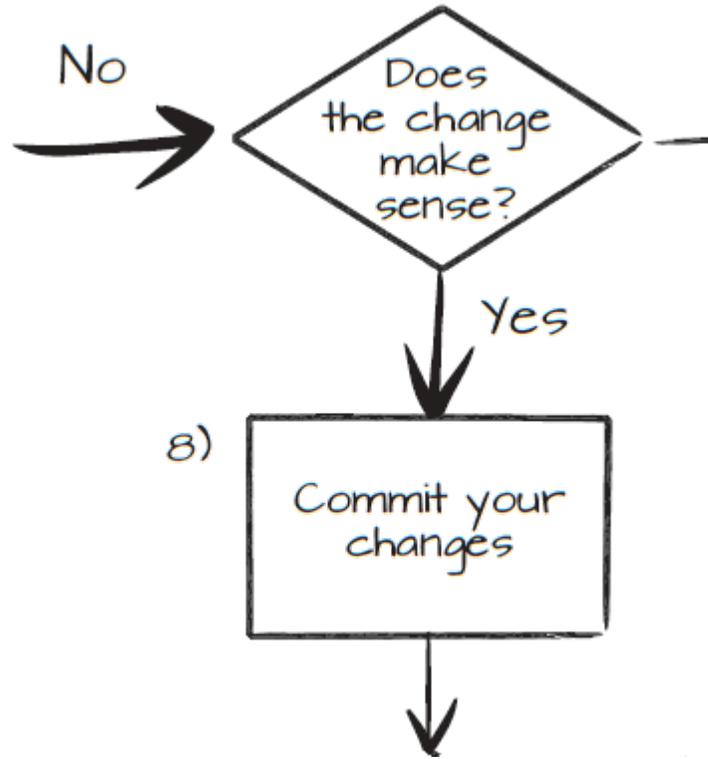


Nächste Vorbedingung auswählen und zu Schritt 2.

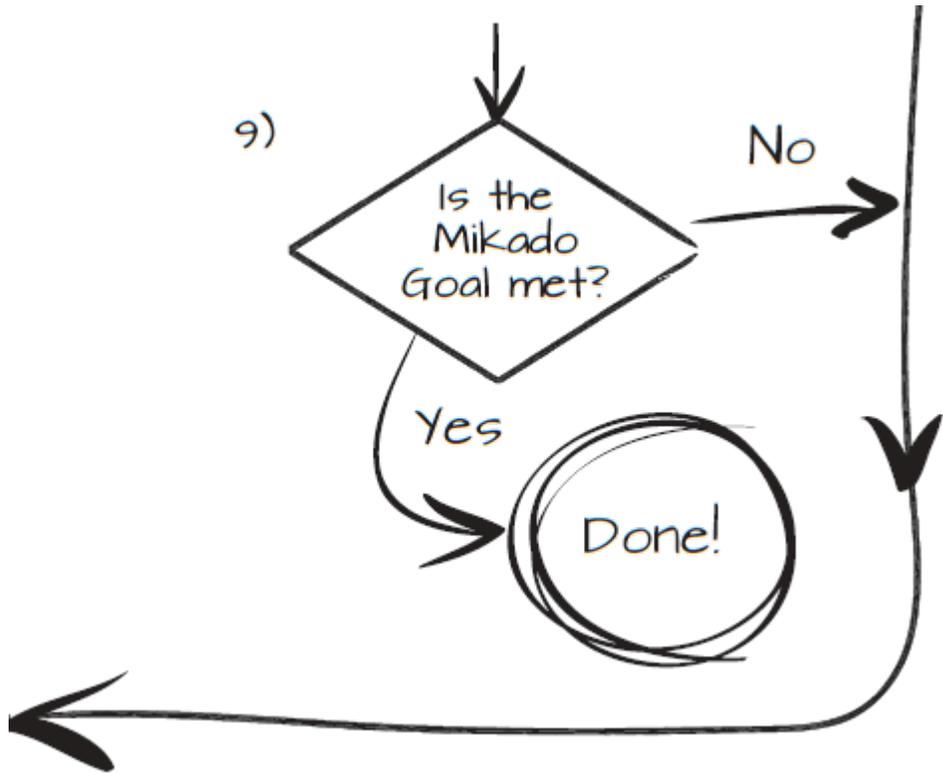
Immer mit sauberem System starten.

Nächste Vorbedingung naiv implementieren ...





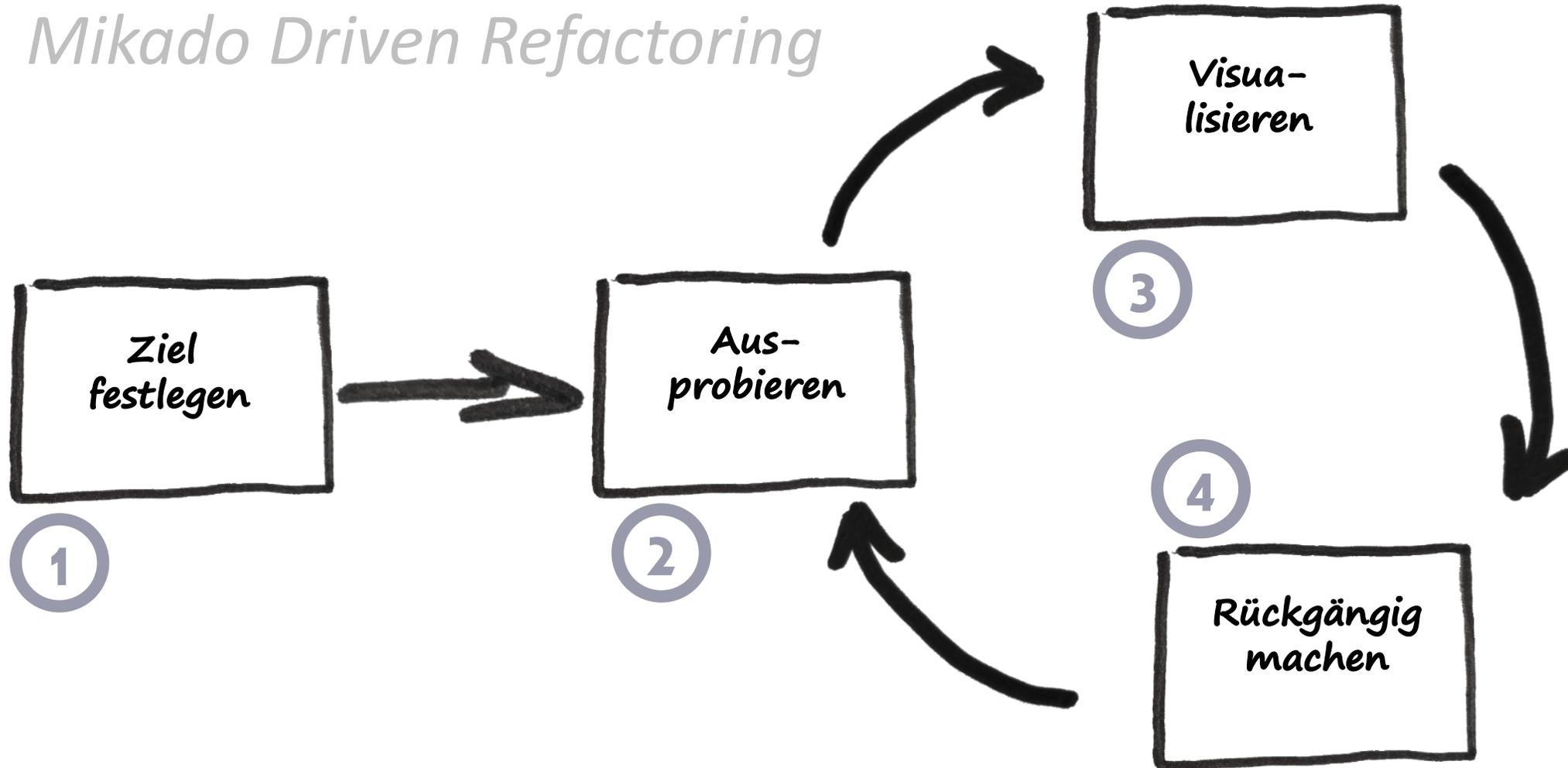
***Vorbedingung abhaken, wenn keine Fehler mehr.
Committen, wenn es Sinn ergibt.***



Mikado-Goal und alle Vorbedingungen erfüllt?

Fertig!

Mikado Driven Refactoring



1

Ziel festlegen

=

***Startpunkt der Änderung.
Erfolgskriterium für Ende.***

Mondscheinzeit-
berechnungslogik
verbessern

2

Ausprobieren

=

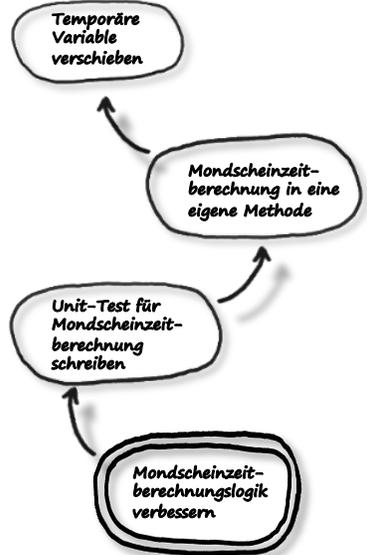
Experimentieren.

Hypothesen prüfen.

Sehen, was kaputt geht.



3



Visualisieren

=

**Ziel und notwendige
Vorbedingungen aufschreiben.
Mikado Graph erstellen.**

4

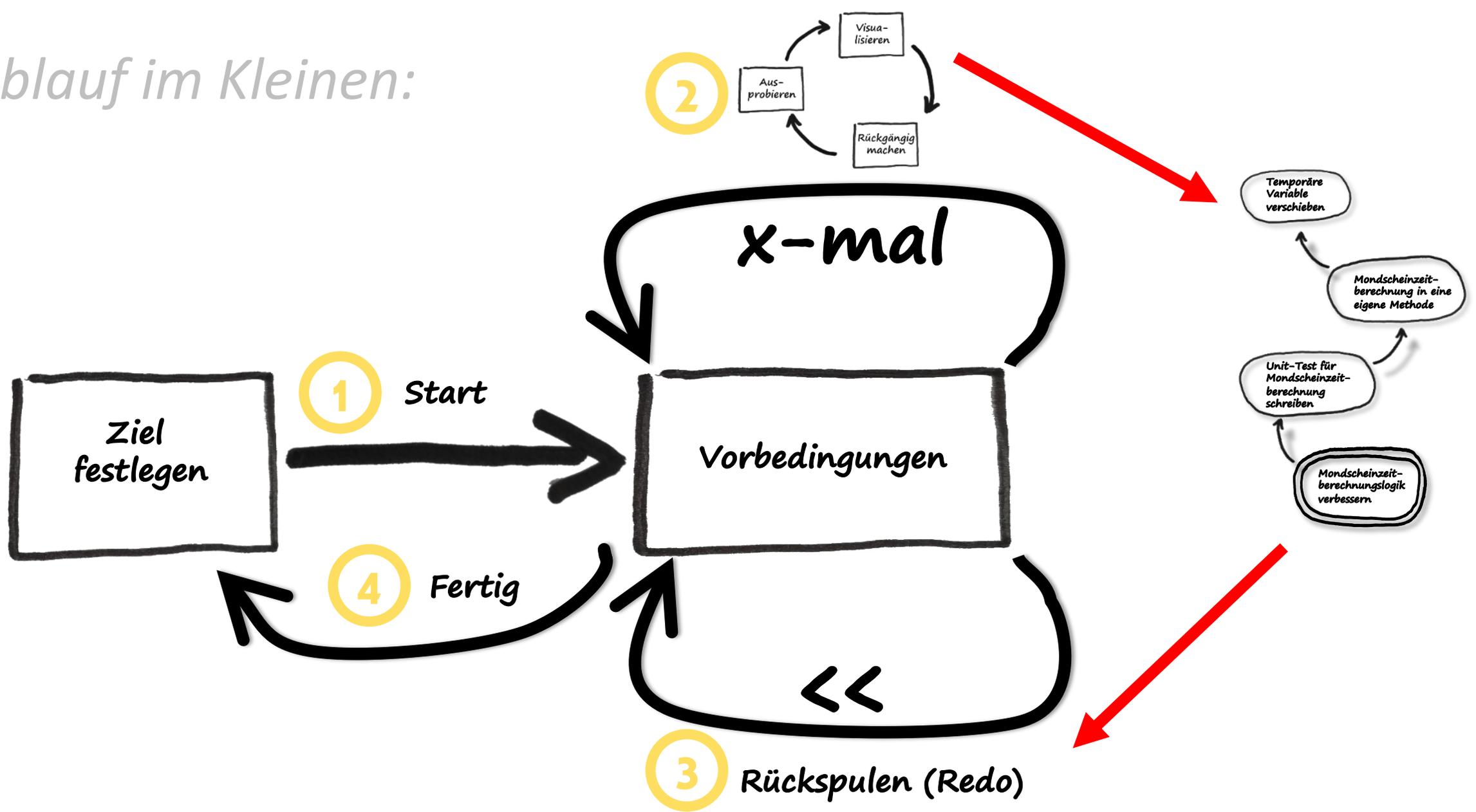
Rückgängig machen

=

***Vorherigen funktionierenden
Stand wiederherstellen.***



Ablauf im Kleinen:



```

package de.oio.refactoring.badtelefon;

public class Kunde {
    double gebuehr = 0.0;
    Tarif tarif;

    public Kunde(int tarifArt) {
        this.tarif = new Tarif(tarifArt);
    }

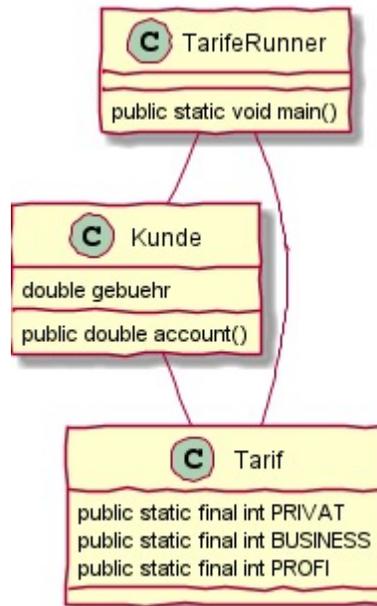
    public void account(int minuten, int stunde, int minute) {
        boolean mondschein = false;
        double preis = 0;

        // Mondscheinzeit ?
        if (stunde < 9 || stunde > 18)
            mondschein = true;

        // Gespraechspreis ermitteln
        switch (tarif.tarif) {
            case Tarif.PRIVAT:
                [...]
        }
        gebuehr += preis;
    }

    public double getGebuehr() {
        return gebuehr;
    }
}

```



```

package de.oio.refactoring.badtelefon;

```

```

import java.util.Arrays;
import java.util.Random;

```

```

public class TarifeRunner {
    public static void main(String args[]) {
        Random random = new Random();
        for(Integer tarif : Arrays.asList(Tarif.PRIVAT, Tarif.BUSINESS, Tarif.PROFI)) {
            System.out.println(String.format("\nVerarbeitung von Tarif %s", tarif));
            Kunde k = new Kunde(tarif);
            [...]
        }
    }
}

```

```

package de.oio.refactoring.badtelefon;

```

```

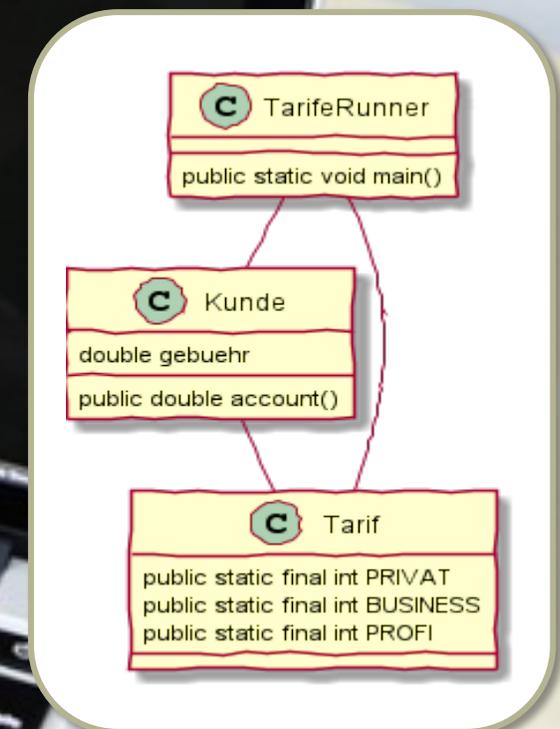
public class Tarif {
    public final static int PRIVAT = 0;
    public final static int BUSINESS = 1;
    public final static int PROFI = 2;

    int tarif = 0;

    public Tarif(int tarif) {
        this.tarif = tarif;
    }
}

```

<https://github.com/sippsack/BadTelefon-Refactoring-Legacy-Code/blob/mikado-dev-initial/doc/mikado-live-coding.adoc>

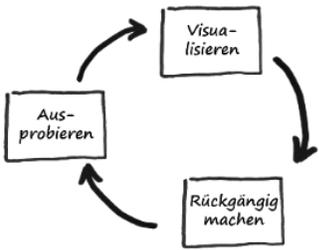


Temporäre Variable verschieben

Mondscheinzeit-berechnung in eine eigene Methode

Unit-Test für Mondscheinzeit-berechnung schreiben

Mondscheinzeit-berechnungslogik verbessern



Zurückspulen
Experimentieren

Zurückrollen
Visualisieren
Experimentieren

Zurückrollen
Visualisieren
Experimentieren

Zurückrollen
Visualisieren
Experimentieren
Mikado-Ziel festlegen





Agenda



- 1 Einstieg
- 2 Legacy Code
- 3 Mikado Methode
- 4 Reale Legacy Projekte**
- 5 Ausblick und weitere Informationen

4

Einfach loslegen?

Es fehlen automatisierte Tests!

Machen wir auch nichts kaputt?



```
# 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)



```
@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



```

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);
    }
}

```

2



3

```

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

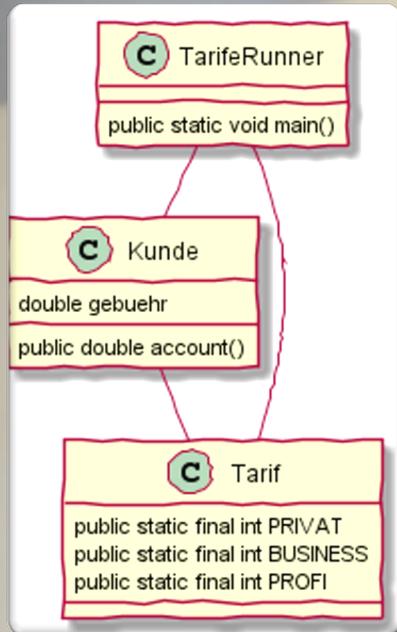
```

4

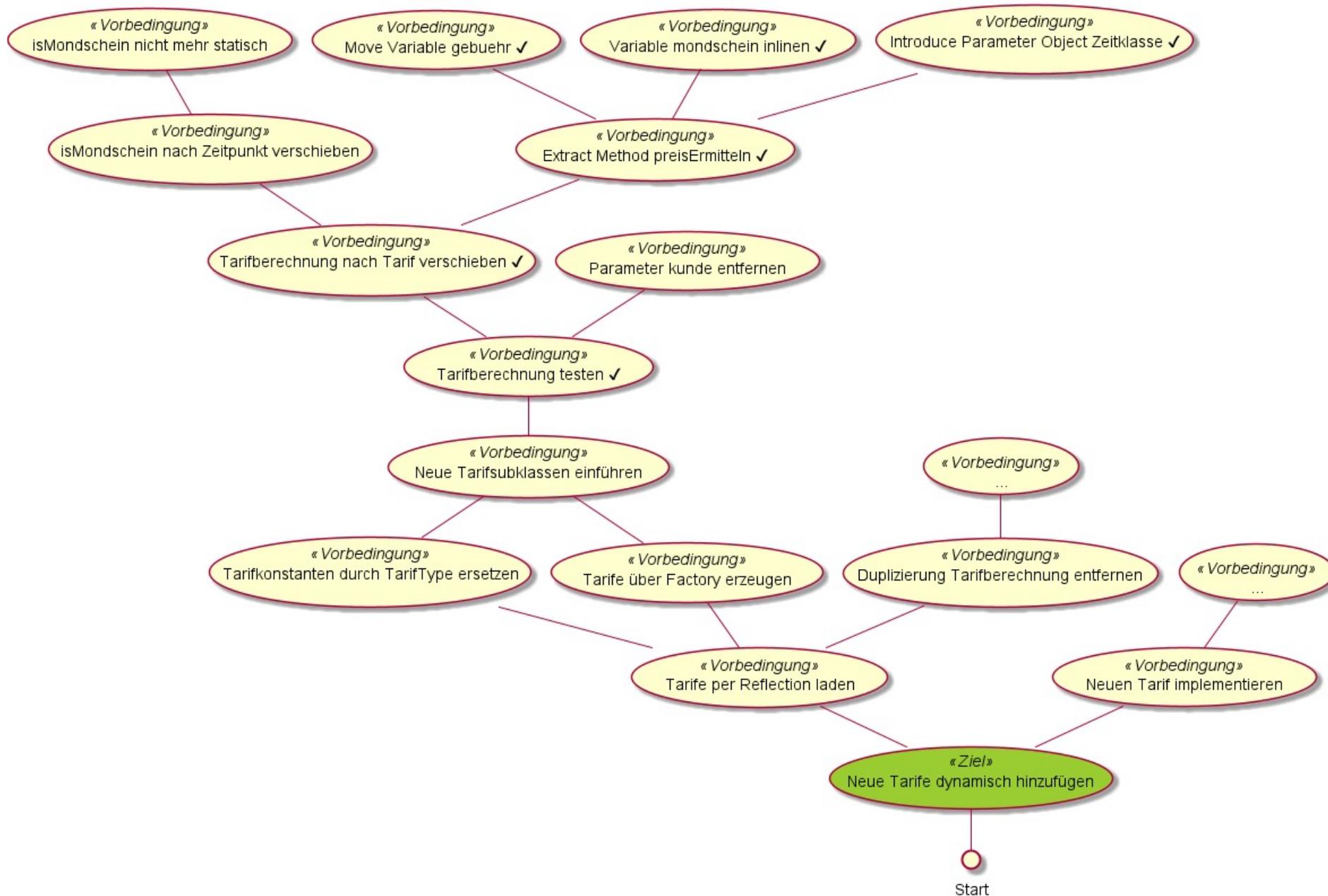


https://entwicklertag.de/frankfurt/2016/sites/entwicklertag.de.frankfurt.2016/files/slides/Bei%20uns%20testen%20lauter%20Affen_0.pdf





```
public class TarifeRunner {
    public static void main() {
        Kunde kunde = new Kunde(10);
        Kunde kunde2 = new Kunde(20);
        Kunde kunde3 = new Kunde(30);
        Kunde kunde4 = new Kunde(40);
        Kunde kunde5 = new Kunde(50);
        Kunde kunde6 = new Kunde(60);
        Kunde kunde7 = new Kunde(70);
        Kunde kunde8 = new Kunde(80);
        Kunde kunde9 = new Kunde(90);
        Kunde kunde10 = new Kunde(100);
        Kunde kunde11 = new Kunde(110);
        Kunde kunde12 = new Kunde(120);
        Kunde kunde13 = new Kunde(130);
        Kunde kunde14 = new Kunde(140);
        Kunde kunde15 = new Kunde(150);
        Kunde kunde16 = new Kunde(160);
        Kunde kunde17 = new Kunde(170);
        Kunde kunde18 = new Kunde(180);
        Kunde kunde19 = new Kunde(190);
        Kunde kunde20 = new Kunde(200);
        Kunde kunde21 = new Kunde(210);
        Kunde kunde22 = new Kunde(220);
        Kunde kunde23 = new Kunde(230);
        Kunde kunde24 = new Kunde(240);
        Kunde kunde25 = new Kunde(250);
        Kunde kunde26 = new Kunde(260);
        Kunde kunde27 = new Kunde(270);
        Kunde kunde28 = new Kunde(280);
        Kunde kunde29 = new Kunde(290);
        Kunde kunde30 = new Kunde(300);
        Kunde kunde31 = new Kunde(310);
        Kunde kunde32 = new Kunde(320);
        Kunde kunde33 = new Kunde(330);
        Kunde kunde34 = new Kunde(340);
        Kunde kunde35 = new Kunde(350);
        Kunde kunde36 = new Kunde(360);
        Kunde kunde37 = new Kunde(370);
        Kunde kunde38 = new Kunde(380);
        Kunde kunde39 = new Kunde(390);
        Kunde kunde40 = new Kunde(400);
        Kunde kunde41 = new Kunde(410);
        Kunde kunde42 = new Kunde(420);
        Kunde kunde43 = new Kunde(430);
        Kunde kunde44 = new Kunde(440);
        Kunde kunde45 = new Kunde(450);
        Kunde kunde46 = new Kunde(460);
        Kunde kunde47 = new Kunde(470);
        Kunde kunde48 = new Kunde(480);
        Kunde kunde49 = new Kunde(490);
        Kunde kunde50 = new Kunde(500);
        Kunde kunde51 = new Kunde(510);
        Kunde kunde52 = new Kunde(520);
        Kunde kunde53 = new Kunde(530);
        Kunde kunde54 = new Kunde(540);
        Kunde kunde55 = new Kunde(550);
        Kunde kunde56 = new Kunde(560);
        Kunde kunde57 = new Kunde(570);
        Kunde kunde58 = new Kunde(580);
        Kunde kunde59 = new Kunde(590);
        Kunde kunde60 = new Kunde(600);
        Kunde kunde61 = new Kunde(610);
        Kunde kunde62 = new Kunde(620);
        Kunde kunde63 = new Kunde(630);
        Kunde kunde64 = new Kunde(640);
        Kunde kunde65 = new Kunde(650);
        Kunde kunde66 = new Kunde(660);
        Kunde kunde67 = new Kunde(670);
        Kunde kunde68 = new Kunde(680);
        Kunde kunde69 = new Kunde(690);
        Kunde kunde70 = new Kunde(700);
        Kunde kunde71 = new Kunde(710);
        Kunde kunde72 = new Kunde(720);
        Kunde kunde73 = new Kunde(730);
        Kunde kunde74 = new Kunde(740);
        Kunde kunde75 = new Kunde(750);
        Kunde kunde76 = new Kunde(760);
        Kunde kunde77 = new Kunde(770);
        Kunde kunde78 = new Kunde(780);
        Kunde kunde79 = new Kunde(790);
        Kunde kunde80 = new Kunde(800);
        Kunde kunde81 = new Kunde(810);
        Kunde kunde82 = new Kunde(820);
        Kunde kunde83 = new Kunde(830);
        Kunde kunde84 = new Kunde(840);
        Kunde kunde85 = new Kunde(850);
        Kunde kunde86 = new Kunde(860);
        Kunde kunde87 = new Kunde(870);
        Kunde kunde88 = new Kunde(880);
        Kunde kunde89 = new Kunde(890);
        Kunde kunde90 = new Kunde(900);
        Kunde kunde91 = new Kunde(910);
        Kunde kunde92 = new Kunde(920);
        Kunde kunde93 = new Kunde(930);
        Kunde kunde94 = new Kunde(940);
        Kunde kunde95 = new Kunde(950);
        Kunde kunde96 = new Kunde(960);
        Kunde kunde97 = new Kunde(970);
        Kunde kunde98 = new Kunde(980);
        Kunde kunde99 = new Kunde(990);
        Kunde kunde100 = new Kunde(1000);
    }
}
```



Agenda



- 1 Einstieg
- 2 Legacy Code
- 3 Mikado Methode
- 4 Reale Legacy Projekte
- 5 **Ausblick und weitere Informationen**

5

Vorteile

Stabiles System trotz Änderungen.

***Bessere Kommunikation und
Zusammenarbeit.***

Leichtgewichtig und fokussiert.



Wann nutzen?

- ① ***Architektur im Betrieb verbessern.***
- ② ***Brownfield Entwicklung.***
- ③ ***Refactoring Projekt.***



① *Architektur im Betrieb verbessern*

***In kleinen Schritten verbessern.
Im gleichen Branch neue Features
kontinuierlich ausliefern.***



② *Brownfield Entwicklung*

***Häufige Situation.
Existierende Anwendungen
verbessern.***



③ Refactoring Projekt

***Langdauernde Verbesserung.
Eigener Branch.***



Angst vorm Revert:

Reverting scheint wie Wegwerfen.

Aber Mikado-Graph enthält Infos.

Wissensaufbau/-transfer über

System, Domäne, Technologie.



Angst vorm Revert: Änderungen doch aufheben

Stash (Git)

Patch-File (SVN, ...)



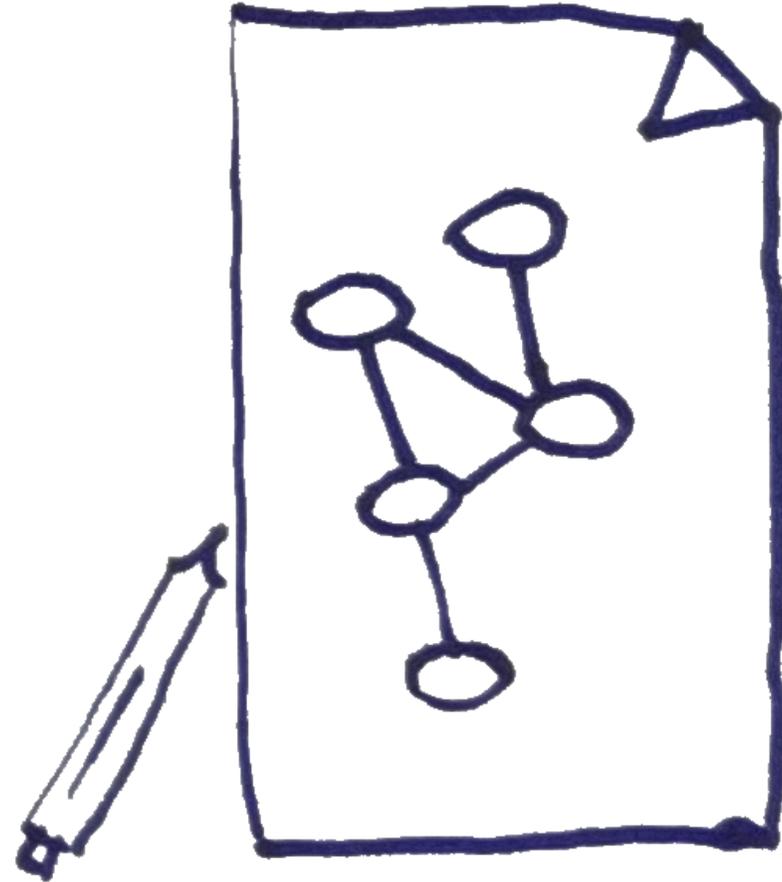
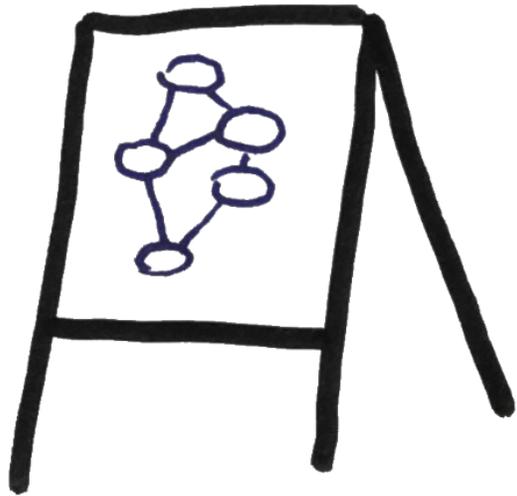
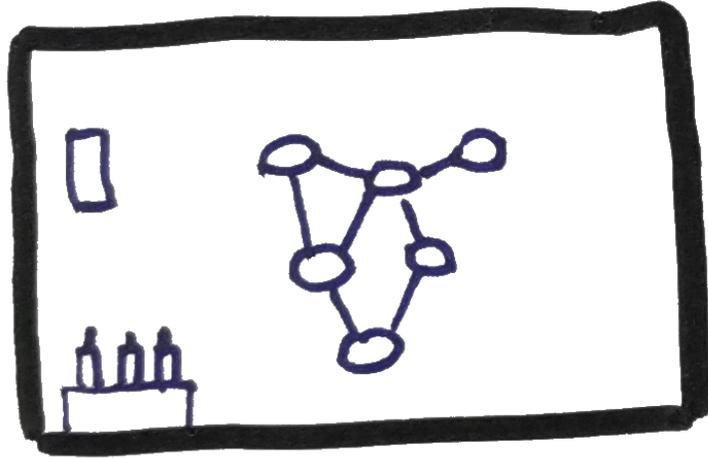
Arbeitsmittel:

① ***Whiteboard/Flipchart/Papier***

② ***Mindmap***

③ ***Visio, yEd***

1



Mikado-Ziel.mm - FreeMind - Mindmapmodus C:\01-presentations\legacycode\Mikado-Methode\Mikado-Ziel.mm

Datei Bearbeiten Ansicht Einfügen Format Navigieren Extras Maps Hilfe

Mikado-Ziel.mm

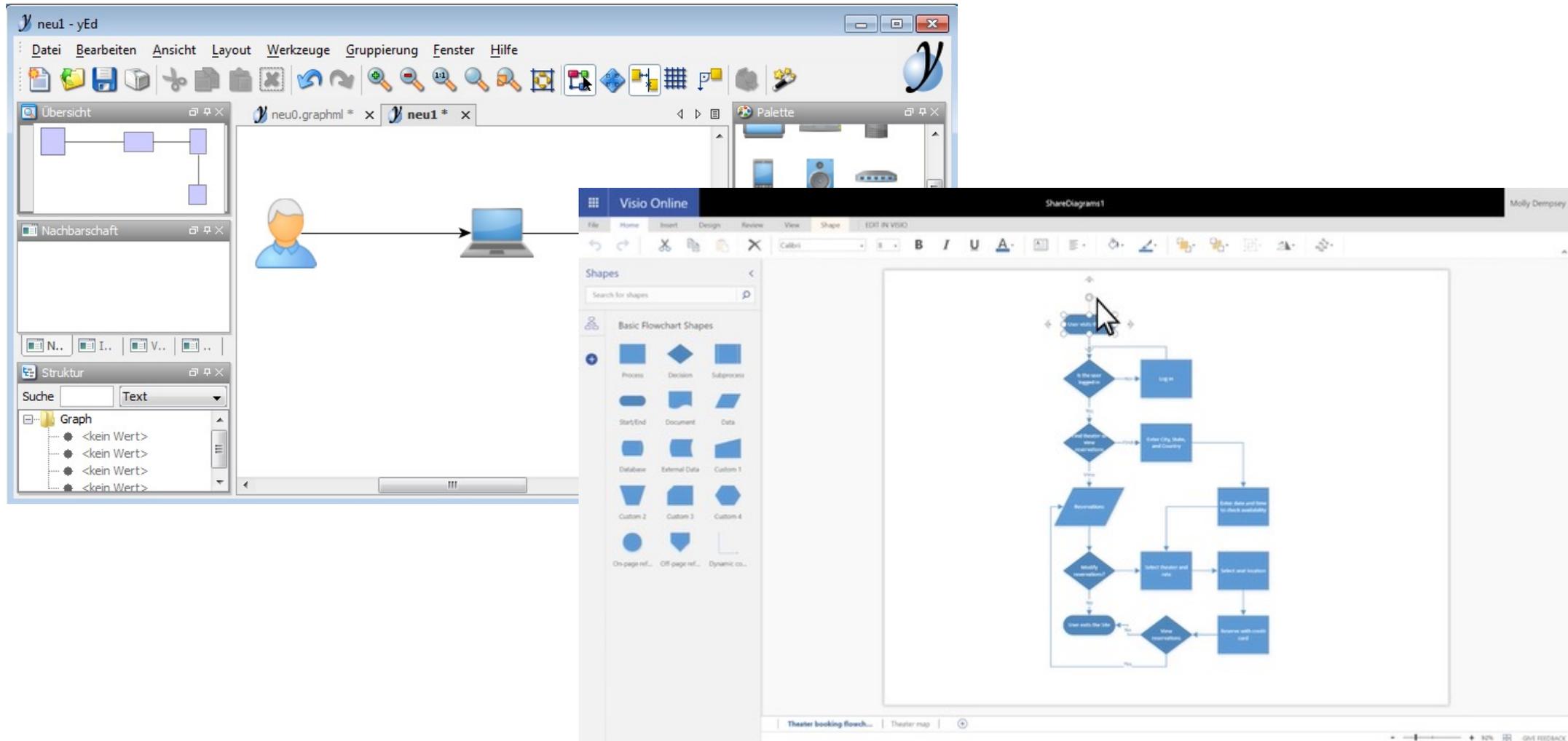
Vorbedingung 2_1 Vorbedingung 2 Mikado-Ziel Vorbedingung 1 Vorbedingung 1_1 Vorbedingung 1_2

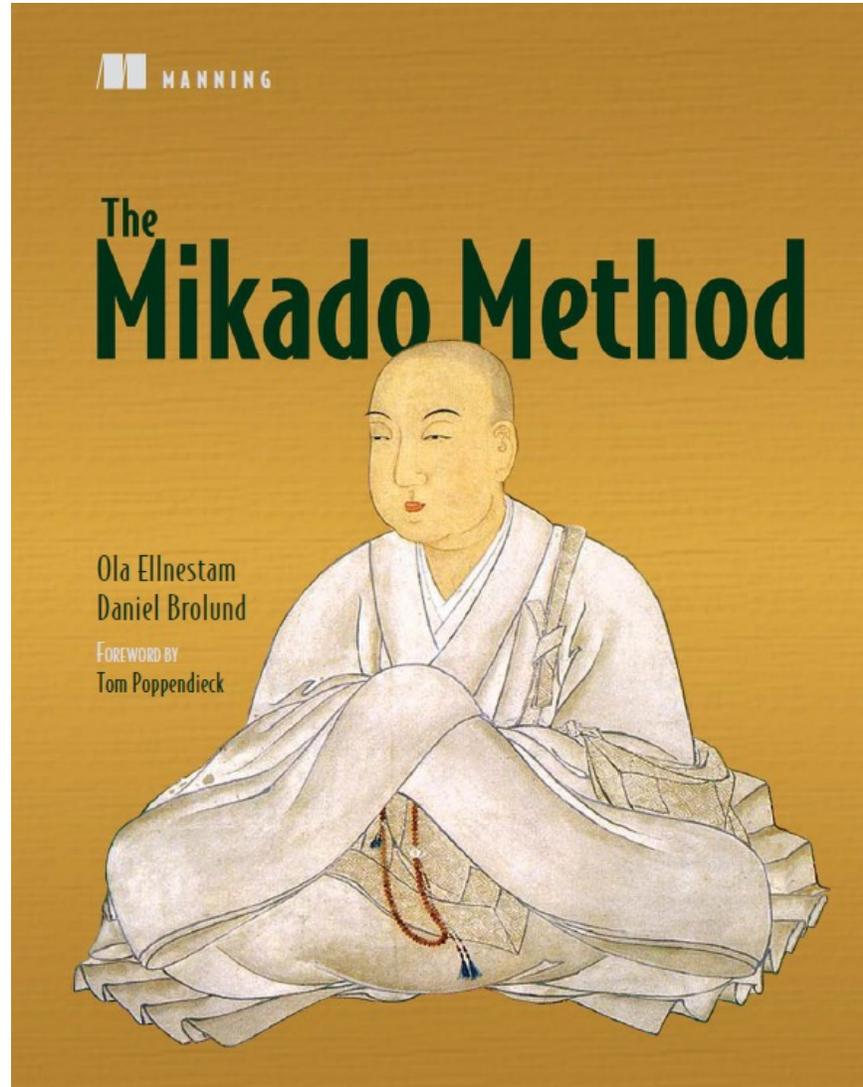
Bearbeiten Format Tabelle Hilfe

SansSerif 11 **b** *i* u T [Icons]

Layout-Ansicht HTML-Code-Ansicht

Mindmap wurde gespeichert.

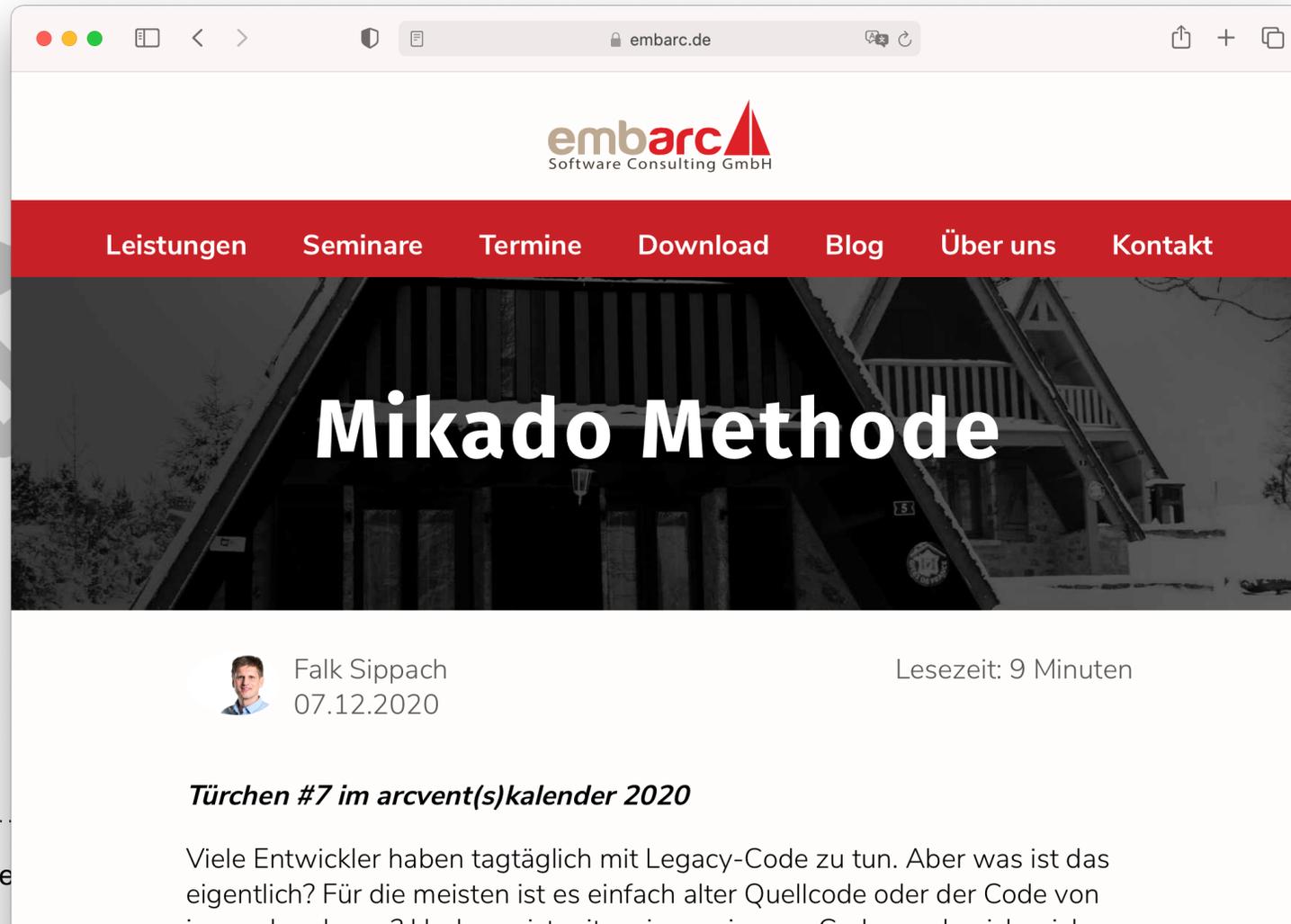




Blog-Beitrag zur Mikado-Methode

Komplexe Refactorings mit der Mikado-Methode in den Griff bekommen.

→ <https://www.embarc.de/mikado-methode/>



The screenshot shows a web browser window displaying the website of embarc Software Consulting GmbH. The browser's address bar shows the URL [embarc.de](https://www.embarc.de). The website's header features the embarc logo and a navigation menu with the following items: Leistungen, Seminare, Termine, Download, Blog, Über uns, and Kontakt. The main content area has a dark background with a large white heading "Mikado Methode". Below the heading, there is a profile picture of Falk Sippach, his name, the date "07.12.2020", and the reading time "Lesezeit: 9 Minuten". The article title is "Türchen #7 im arcvent(s)kalender 2020". The beginning of the article text is visible: "Viele Entwickler haben tagtäglich mit Legacy-Code zu tun. Aber was ist das eigentlich? Für die meisten ist es einfach alter Quellcode oder der Code von".

Folien und Quellcode zum Download

embarc
Software Consulting GmbH

Leistungen | Seminare | Termine | Download | Blog | Über uns | Kontakt

Downloads und Medien

Unsere Folien, Videos, Artikel und Beiträge

Vortragsfolien | Videos | Alle

Hier gehts zu unseren Architekturspickern →

embarc.de/download/



Vielen Dank.

Ich freue mich auf Eure Fragen!



Falk Sippach



fs@embarc.de



@sipp sack



→ [xing.to/fsi](https://www.xing.to/fsi)