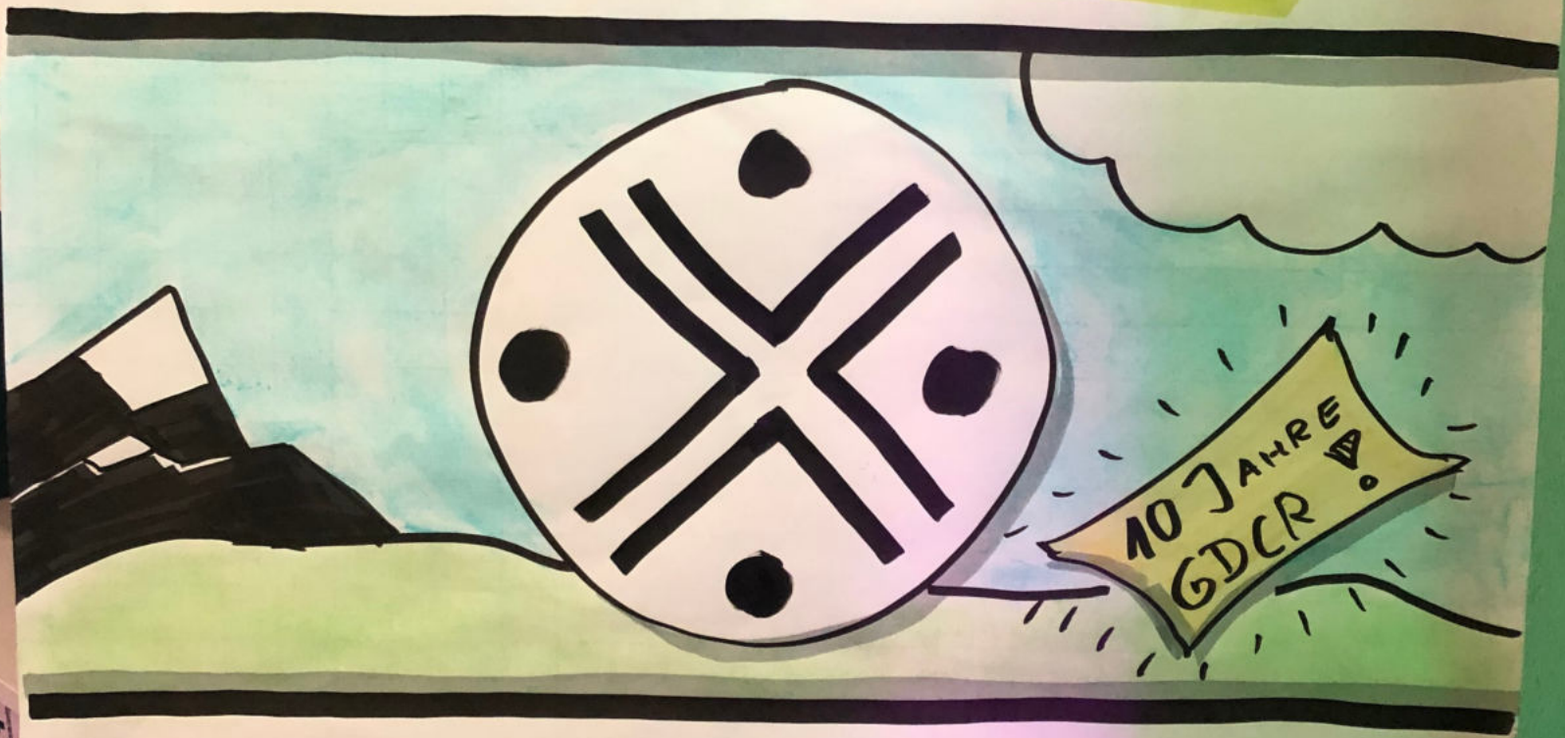
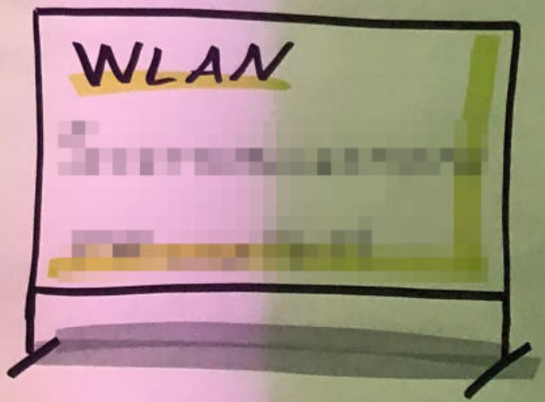
Willkommen beim

GDCR 2019



bei der liseo GmbH

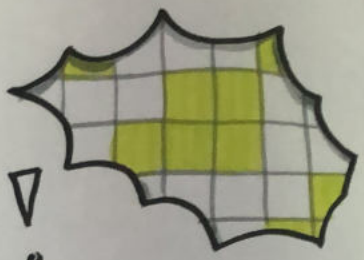


Agenda

	Lounge 2.0G	301 3.0G	GRB 3.0G	404 3.0G
9:00	Intro			
10:00			TDD	TDD
11:00		TDD	Ping Pong	Frei
12:00	Pause			
13:00		Ping Pong	Muted Ping Pong	Frei
14:00		No Primitives	Immutability	Frei
15:00		Max. 4 Zeilen	Every Method is void	Frei
16:00		Closing Circle		
17:00	After Glow			

Session ?

Aufgabe: Programmiere Game of Life!



Nutze dabei eine oder mehrere Einschränkungen (Constraints)

5 Min Pause

10 Min. Retro

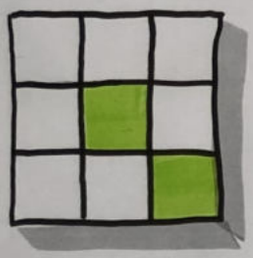
Raum: ~~301~~ Flur

Stop!
Delete your code
Stand up!

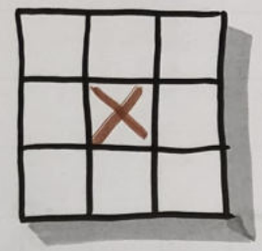
45 Min.
Code

Game of Life Rules

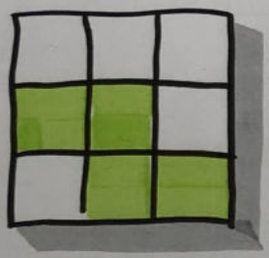
Generation X ⟶ Generation X + 1



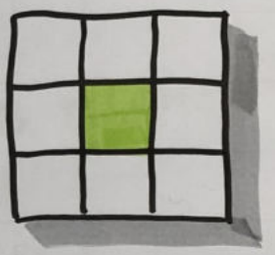
< 2
under population

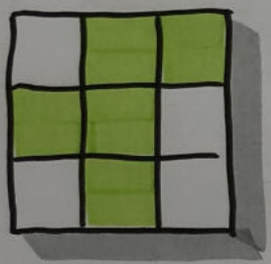


†

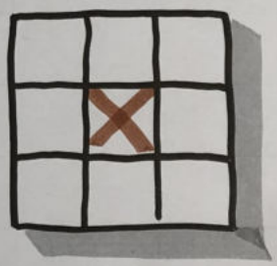


= 2, 3
survival

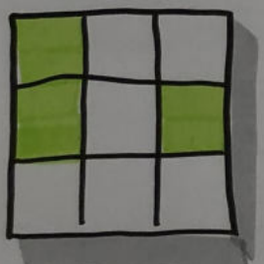




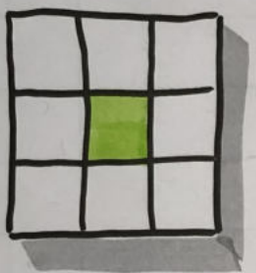
> 3
over crowded



†



= 3
birth



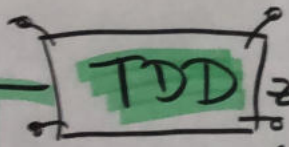
*

Session #1

- viele 2D-Arrays
  - ↳ ∴ man geht auch alle toten Zellen durch
  - ↳ Randproblem

- lebende Zellen = eine Menge
  & Zellen kennen ihre Nachbarn

n-dimensional: Nachbarn steigen

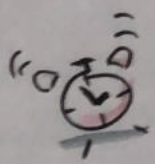


zum ersten Mal:

Umdenken, sehr stark

„fühlt sich nicht falsch an"

...obwohl man erst etwas falsches implementiert.



„Moment, ich habe ja noch gar keine Klasse!"

„...hat erst mal länger gedauert."

„Code sieht ganz anders aus."

Session #2

2. Session ging besser, TDD war einfacher

„mit Ping Pong kann man besser nach TDD arbeiten"

Für eine Person neue Sprache: Ping Pong war gut zur Anleitung.

mal 'ne neue Sprache ausprobiert (Kotlin),

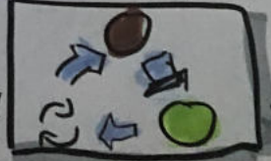
„Ich will ja nur public-Methoden testen. Die privaten Methoden helfen mir ja nur."

Business-Core ist es ja die 8 Nachbarn zu bestimmen.

Zu dritt Ping-Pong: „Wir haben Rundlauf gespielt."

„Man hat immer 2 Leute, die einen zutexten"

Beim Ping Pong ist es interessant, dass man sich Designs noch nicht klar ist, wenn man den Test schreibt.



Session #3

3. Session
3. Umgebung
3. Programmiersprache

= 

Ping Pong eher durch
Laptop hin- und herschieben.

Muted Ping Pong: erstmal implizite Annahmen
: statt refactoring: neue Tests,
die neue Features verlangen

Q: Was, wenn der Test sofort grün ist?

A: Neuen Test schreiben!
A: Im Constructor 'ne Exception schmeißen
A: Der "zu früh kondite" Test hat gar kein neues Feature verlangt.

Durch Ping-Pong entstanden sehr gezielte Tests!

Beim Muted Ping Pong kannten wir uns gar nicht – es hat aber trotzdem geklappt

Session #4

No Primitives

- schwer! Wie bekomme ich Werte von A nach B?
- Klassen statt Zuständen
- ... wir konnten equals(.) nicht umsetzen
- „Wir waren noch nicht an dem Punkt, wo es die Entwicklung einfacher macht."
- Erzeugt starke Typsicherheit in APIs

Immutables only. Outside-In Entwicklung

Mit neuer Sprache (Kotlin) sehr herausfordernd

„Wir haben das schon mimer gemacht ..."
„... war nicht so schwer"

TDD – as you meant it!

Man kommt erst spät dazu, etwas zu kapseln.

Baby steps

= Ping-Pong mit sehr kleinen Einheiten
Aber sehr gut vorangekommen!
Kann man empfehlen.

1... 2... 3...

Session #5

Max 4 Zeilen: Am Anfang easy, es geht sogar mit nur einem Statement

„... gut nachrollziehbar"

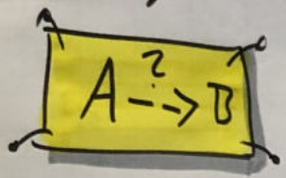
Lesbarkeit hoch, Testbarkeit gleich

x2 - x3 Methoden

Auch Testmethoden sind Methoden

Benennung ist der Schlüssel

Every Method is void:



„Wie schiebe ich Werte von A nach B?
In den Tests ein Problem!

Arbeiten mit Callbacks & Lambdas

„Ruby ist dafür scheiße" 😀

Observer - Pattern

Closing Circle

1. Was habe ich heute gelernt?
2. Was hat mich heute überrascht?
3. Was werde ich ab morgen anders machen?


TDD ist ein zweischneidiges Schwert!
Constraints können helfen Gewohnheiten zu hinterfragen
Gute Entwicklung zeichnet sich durch mehr aus als die Kenntnis dieses Frameworks, Sprachen, Paradigmen.
TDD as if you meant it + Outside In is hard.
- Ping Pong spielen
- Kotlin hardson
- Inside-out vs. Outside-in
Mit mehr Leuten programmieren und mit Pair/Mob Programming macht Coding tatsächlich Spaß!
1
1. etwas Kotlin TDD in der Praxis, vor allem Ping Pong
1]
Viele Wege führen zum Ziel.
TDD prevents untestable code
Tests sind gut
Kotlin really is nicer than Java
Pair-Programming can be very relaxed
Jeder geht auf verschiedene Weise an Probleme heran. Test helfen die Rahmenbedingungen für sich selbst zu schaffen
3. Pair-Programming (war aber ok, gepennt)
3]
3
- TDD
- Pair Programming
- Team Kotta neue Ideen ausprobieren
Jede Session ist ganz anders
Zeit ist knapper als man denkt
2]
Was „TDD as if you meant it" bedeutet. (Gilt dann auch für 1)
-Ping Pong Kennenlernen
-Kotlin sehr interessant
Spaß ++ nette Leute
2
Martin macht auch schöne Folien
It's fun to code with people you don't know (yet)
Code Veranstaltungsvierten Dank?
45 Minuten sind schnell vorbei
Die verwendete Sprache ist Egal
Wie unterschiedlich der Alltag aussieht. (Frameworks/Sprachen das Denken beeinflusst)