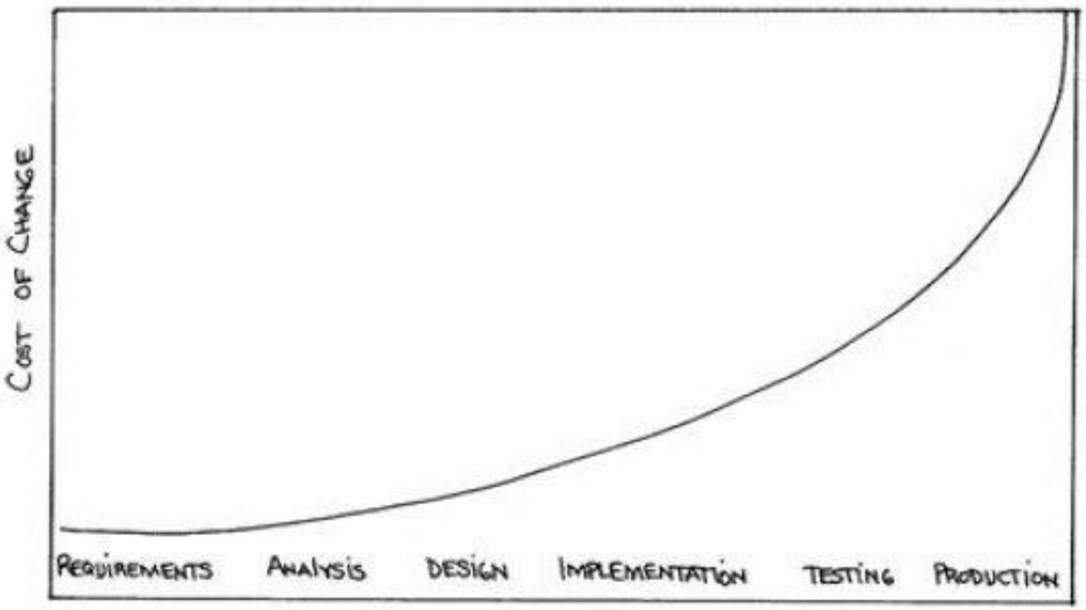
Test -Driven Development

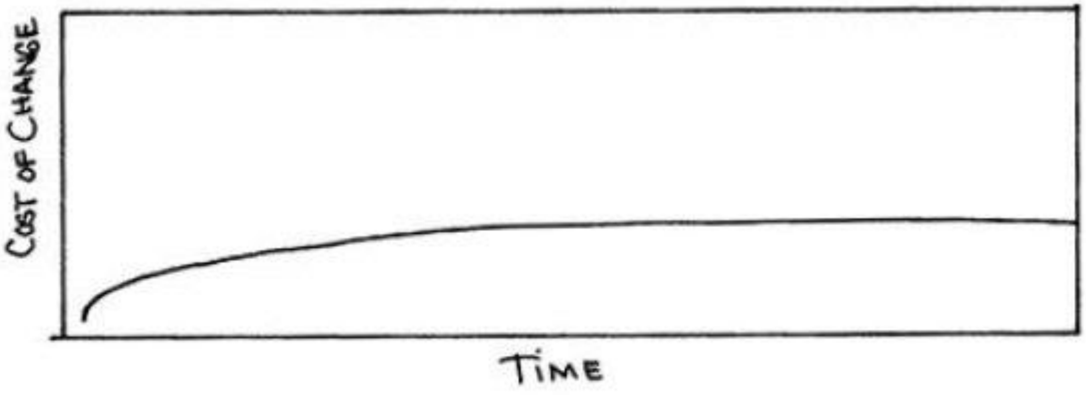
by Coders TUG

Cost of Change

Barry Boehm



Kent Beck

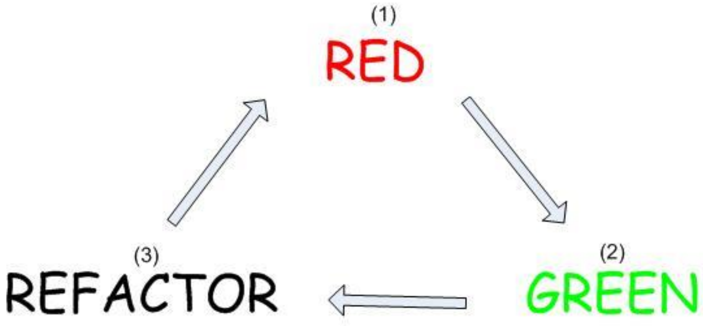


Programming Style Values

The best programs offer many options for future extension, contain no extraneous elements, and are easy to read and understand.

- Communication: Code is primarily means of communication.
- Simplicity: Eliminate excess complexity. Apply simplicity at all levels.
- Flexibility: Programs should be flexible in the ways they change, they should make common changes easy or at least easier.

TDD Mechanics



1. Write a test.
2. Make it compile.
3. Run it to see that it fails.
4. Make it run.
5. Remove duplication.

The **different phases** have **different purposes**. They call for **different styles of solution**, different aesthetic viewpoints. **The first three phases need to go by quickly**, so we get to a known state with the new functionality. We can commit any number of sins to get there, because speed trumps design, just for that brief moment.

Now I'm worried. I've given you **a license to abandon all the principles of good design**. *[cut]* The cycle is not complete. A four-legged Aeron chair falls over. **The first four steps of the cycle won't work without the fifth. Good design at good times. Make it run, make it right.**

There, I feel better. **Now I'm sure you won't show anyone except your partner your code until you've removed the duplication**.

Kent Beck, Test-Driven Development by Example

Simple Design

The team keeps the design exactly suited for the current functionality of the system.

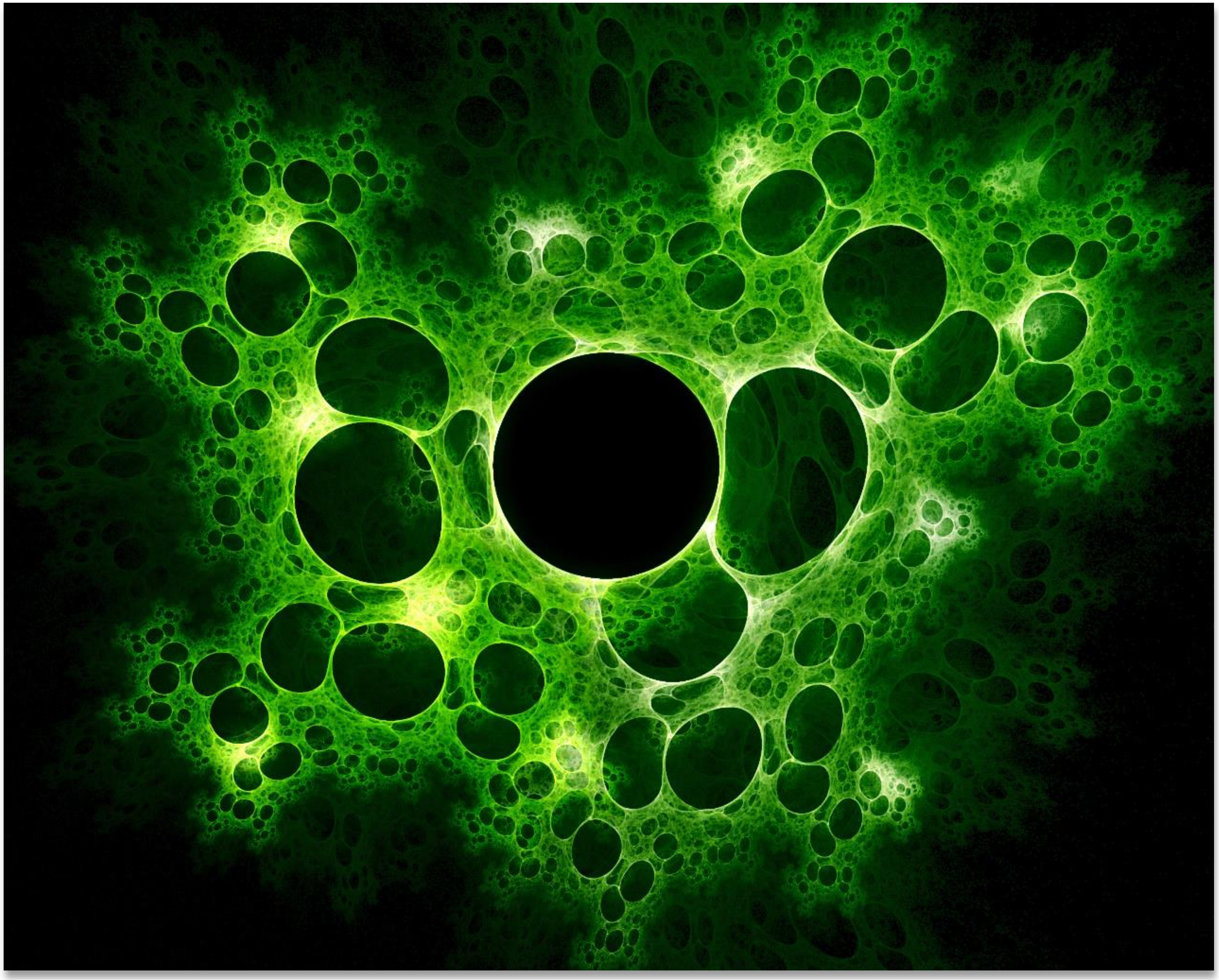
1. Passes all the tests
2. Contains no duplication
3. Express developer intent
4. Contains as little code as possible

(In this order)

“TDD is intended to help programmers take more responsibility for the quality of their work.”

Kent Beck

Clean Code That Works



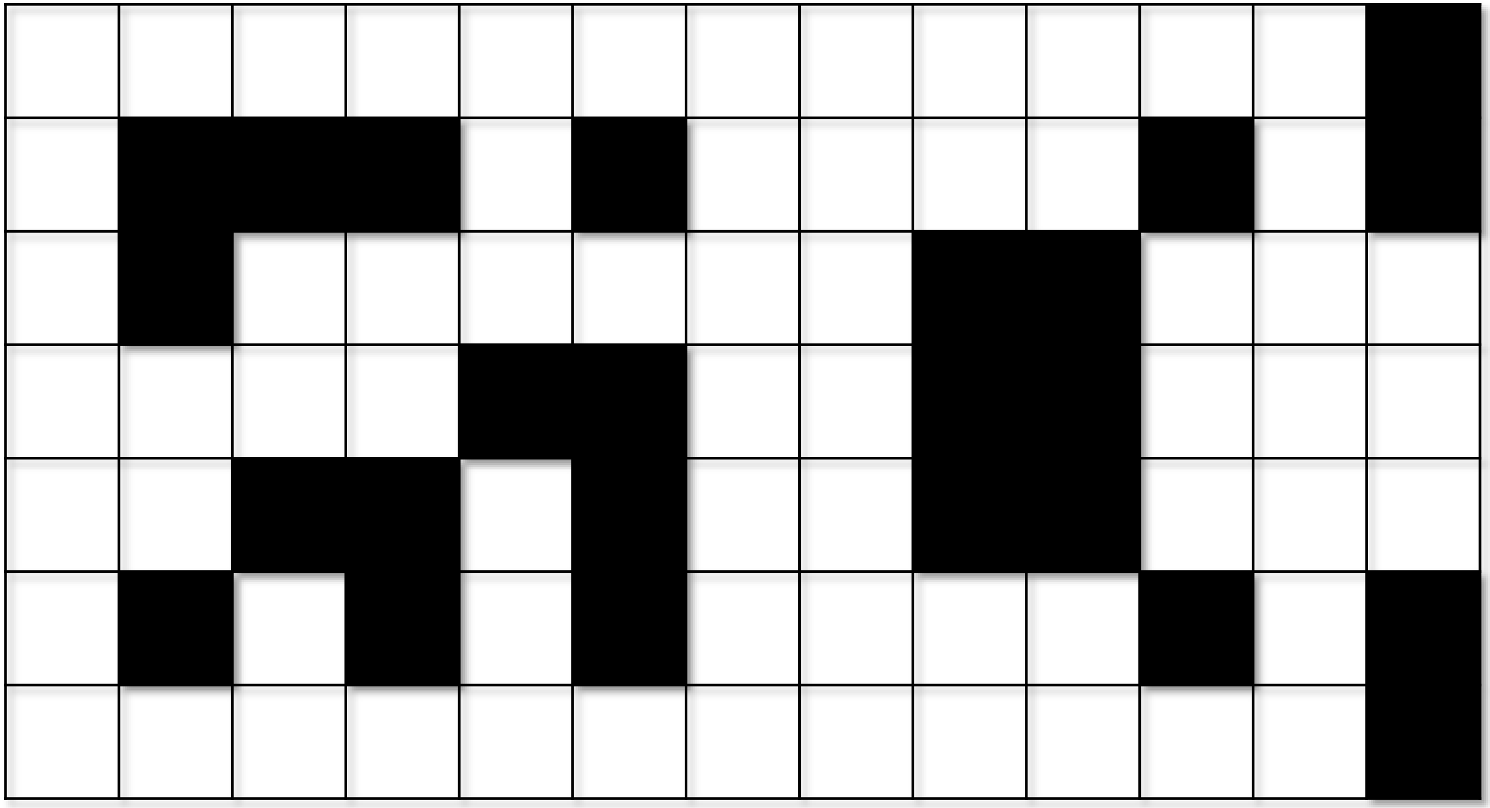
The Game of Life

The Game of Life is a **cellular automaton** devised by the British mathematician **John Conway** in 1970.

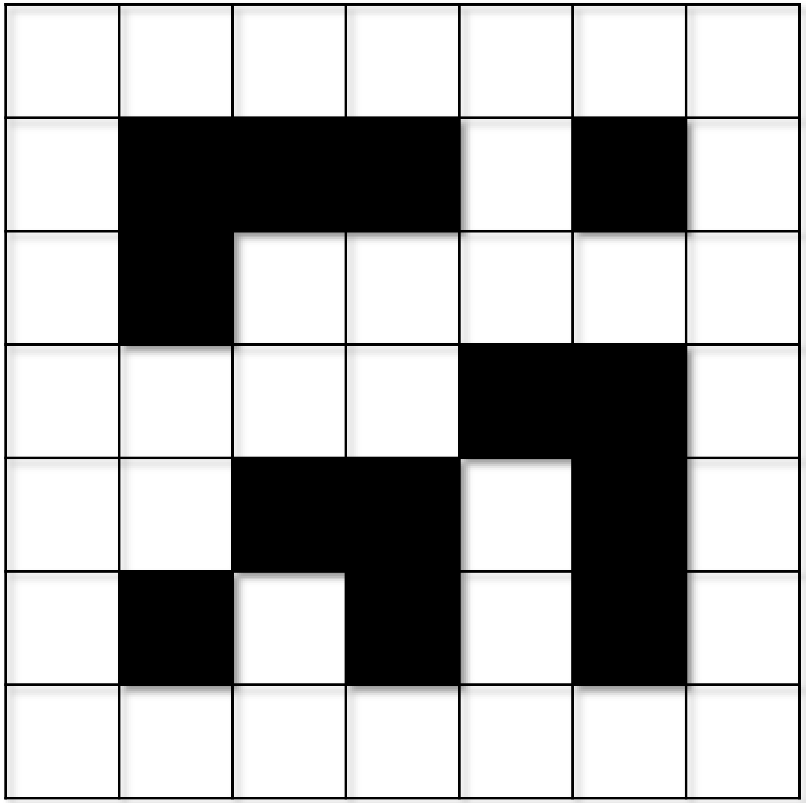
It’s a **zero-player game**.

You **create an initial state** and **watch how it evolves**.

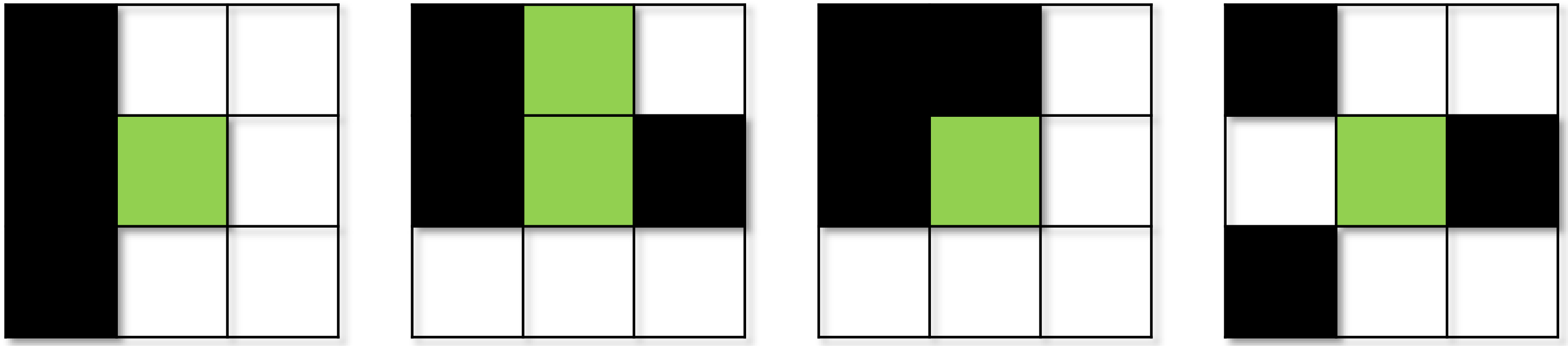
The **universe** Game of Life is an infinite 2D grid of square **cells**, each of which is in one of two possible states, **alive** or **dead**.



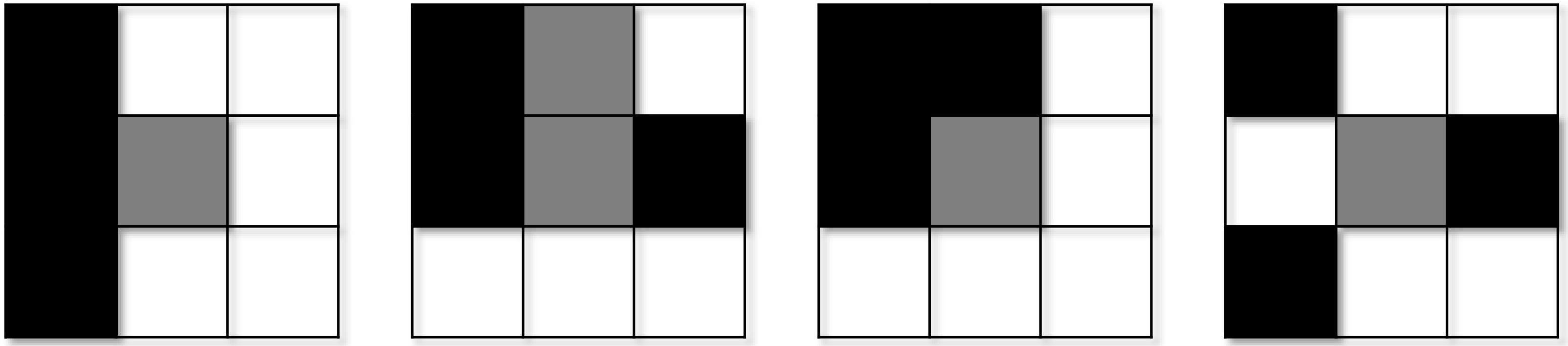
At each *generation*, every cell interacts with its ***eight neighbours***, following ***three rules***.



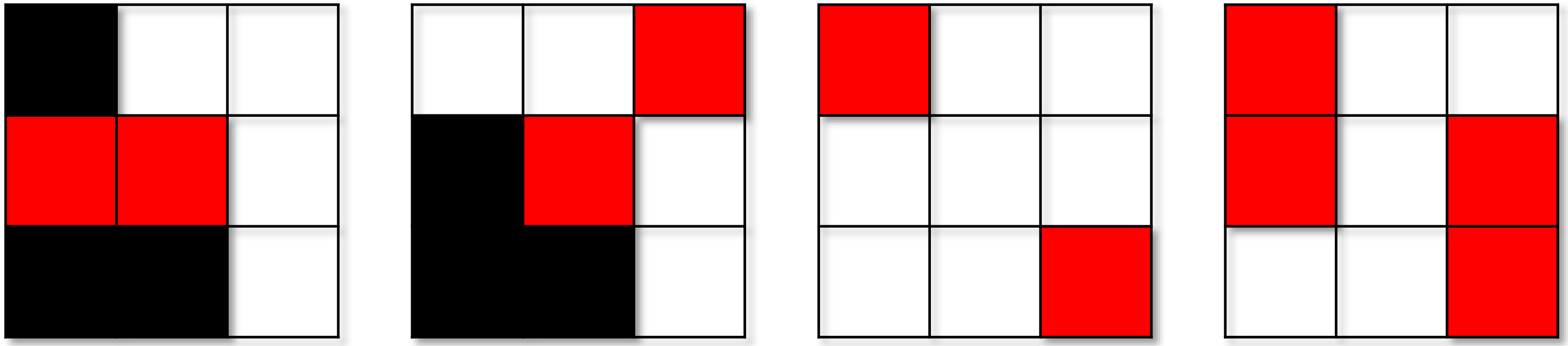
Any **dead** cell with **exactly three live neighbours** **becomes a live** cell, as if by reproduction.



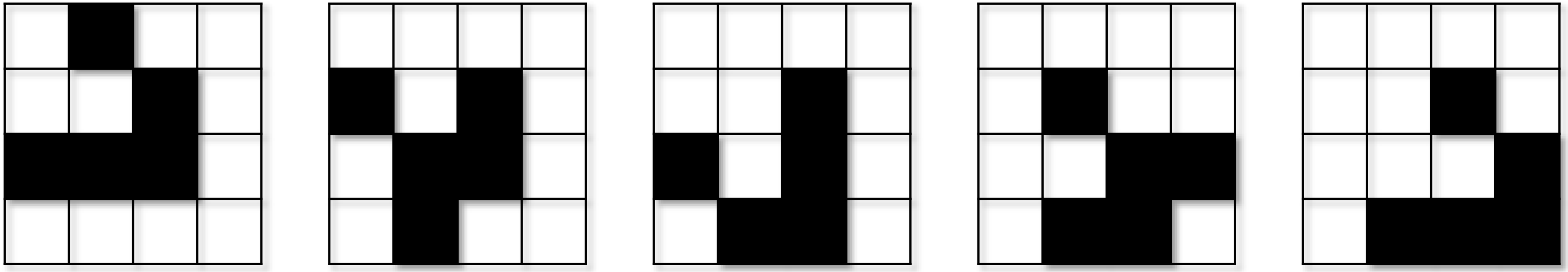
*Any live cell with **two or three live neighbours** lives on to the next generation.*



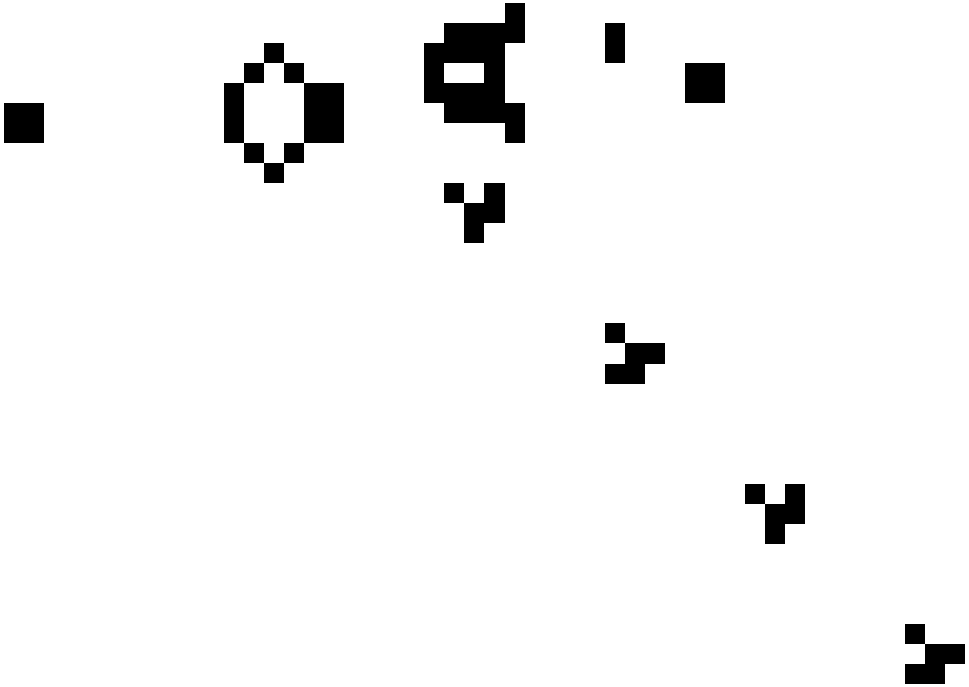
Otherwise, the cell dies from either loneliness or overcrowding.



Depending on how it's seeded, the game board exhibits remarkable, very lifelike, behavior. Like a glider.



The Game of Life demonstrates *emergent behavior*.
The *behavior of the system* as a whole *can't be predicted* solely *by looking at the behavior of the single objects* that comprise the system.

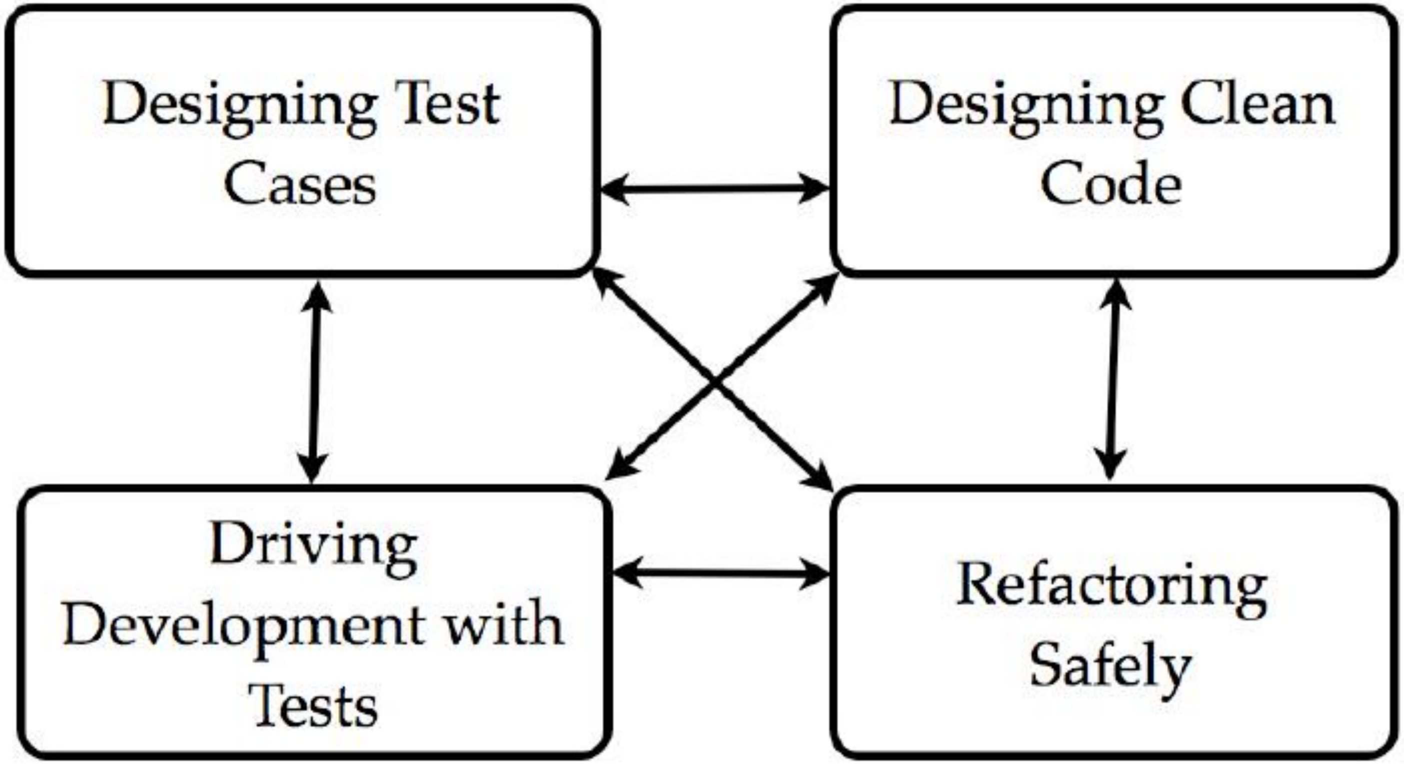


Learn TDD

“TDD doesn't drive good design. TDD gives you immediate feedback about what is likely to be bad design. If a test is hard to write, if a test is non-deterministic, if a test is slow, then something is wrong with the design.”

Kent Beck

Multifaceted Skill



Designing Test Cases

Designing Clean Code

Driving Development with Tests

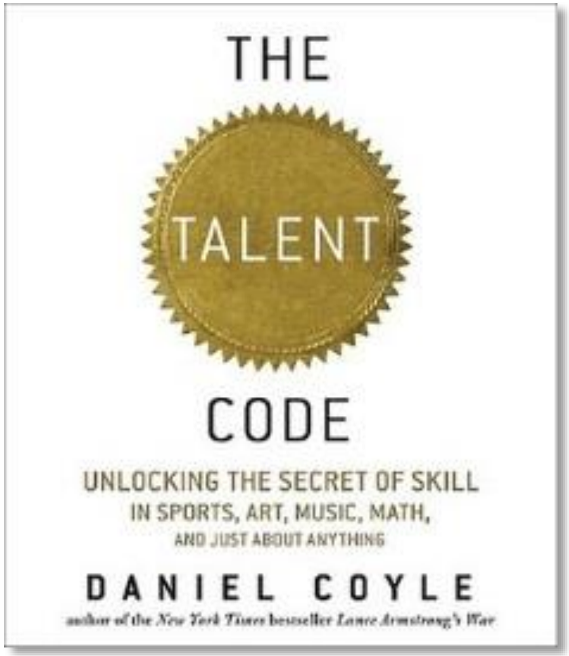
Refactoring Safely

Study

Experiment

Pair Programming

Deliberate Practice

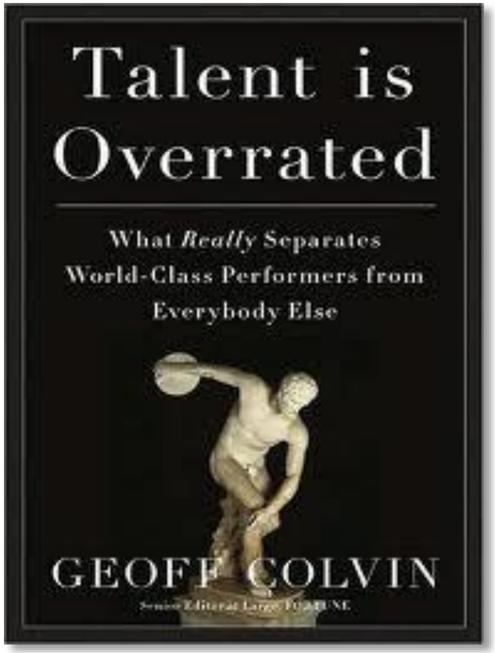


THE TALENT CODE

UNLOCKING THE SECRET OF SKILL IN SPORTS, ART, MUSIC, MATH, AND JUST ABOUT ANYTHING

DANIEL COYLE

author of the *New York Times* bestseller *Lance Armstrong's War*



Talent is Overrated

What *Really* Separates World-Class Performers from Everybody Else

GEOFF COLVIN

Senior Editor at Large, FORTUNE

*The secret is life-long period of deliberate effort to improve performance in a specific domain. The secret is what researcher calls **Deliberate Practice**.*

Code Kata

Bei Nr. 15: Schlag mit rechten Arm in linke Handfläche, dabei rechten Fuß in linke Kniekehle
Achtung:
Techniken 5-8, 9-12 und 25-28 sind genau gleich!
Der gesamte Bewegungsablauf wird aus drei [illegible] Blickwinkeln dargestellt. Bitte genau studieren!
© by A. Flüger
Empi
(früher: WANSHU)
燕飛
„Yoi!"
Linke Faust vor dem Solar-Plexus
Füße links, rechts zusammen!
Rechte Faust senkrecht in linker Handfläche
Gedan-Barai
Rechte Faust auf linke Faust
Kagi-Tsuki
Gedan-Barai
Age-Tsuki (als Block)
Greifen (oder nur Hand) strecken!
weit nach vorne abstoßen!
Greifen
Age-Tsuki
Blick in linke Handfläche
Aufrichten und Gedan-Barai nach vorne
Gedan-Barai (nach hinten)
weit und tief nach vorne springen!
Gedan-Barai
Gedan-Barai
Tsuki
Kosa-Dachi
Ausholbewegung
Tate-Shuto-Uke (kann auch schnell ausgeführt werden!)
Rentsuki (Doppelstoß)
Umsetzen nach links
Age-Tsuki
rechten Fuß zurück!
linken Fuß vor!
Kokutsu-D. Gyaku-Tsuki
Shotei-Uke
Über Nekoashi-Dachi zu Nr. 34. Ausholbewegung der Hände beachten!
Oberkörper und hinteres Bein eine Linie! (Gleich wie Nr. 9 und 14!)
weit und tief nach vorne abstoßen!
Wendung
Schritt rechts vor
„Yame!"
Nr. 33-36: Shotei-Uke (Handballenblocks) langsam und mit zunehmender Spannung
Gleiten!
Gleiten!
Sprung mit Drehung um 360°. Füße hochziehen dabei und ausholen zum Shuto-Uke
Füße links, rechts auseinander!
Dauer etwa 40 Sek.!
Techniken 34, 35 und 36 flüssig und schnell hintereinander!
Zeichenerklärung
= starkes KIME, eine Sekunde Pause
= nach dem KIME gleich weiter
= langsame Bewegung zum vollen KIME
= langsames Entspannen vom vollen KIME
= Zwischenzeit, keine eigene Technik
X = zwei Sekunden Pause
● = KIAI
→ vollendete Bewegung
⇒ Bewegung im nächsten Moment
Kiba-Dachi Vorgleiten und mit beiden Händen greifen
Kokutsu-Dachi mit Gedan-Barai Gleiten!
Bei 35 Oberkörper etwas nach vorne neigen!
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37

Dave Thomas,
Pragmatic Programmers

Coding Dojo

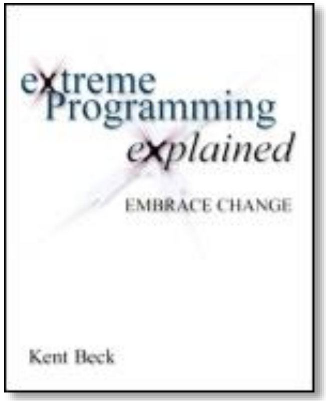
Coderetreat

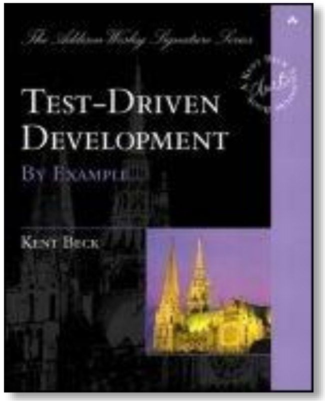
“I'm not a great programmer, I'm just a good programmer with great habits”

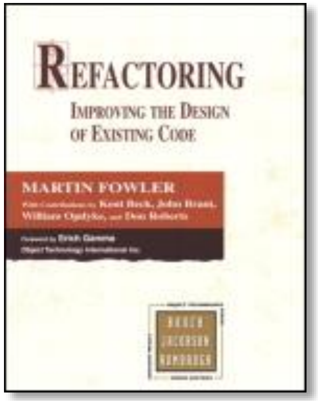
Kent Beck

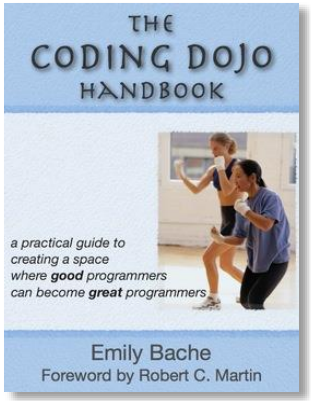
Resources

- http://codingdojo.org/cgi-in/wiki.pl?KataCatalogue
- http://codekata.pragprog.com/
- RIP TDD – By Kent Beck
- Programming Like Kent Beck








THE
CODING DOJO
HANDBOOK
a practical guide to creating a space where good programmers can become great programmers
Emily Bache
Foreword by Robert C. Martin

Matteo Baglini

Freelance Software Developer &
Tecnical Coach

Coders TUG & DotNetToscana
Co-Founder

@matteobaglini

matteo.baglini@gmail.com

CODERS
tuscany user group

Learning Through Sharing

Coders Tuscany User Group è una ***community di sviluppatori appassionati*** *che credono nel collaborative learning.*

Il nostro ***obiettivo*** *è creare un* ***network di coders*** *che desiderino alimentare la propria passione condividendo le proprie conoscenze ed esperienze, sperimentando,* ***imparando insieme****.*