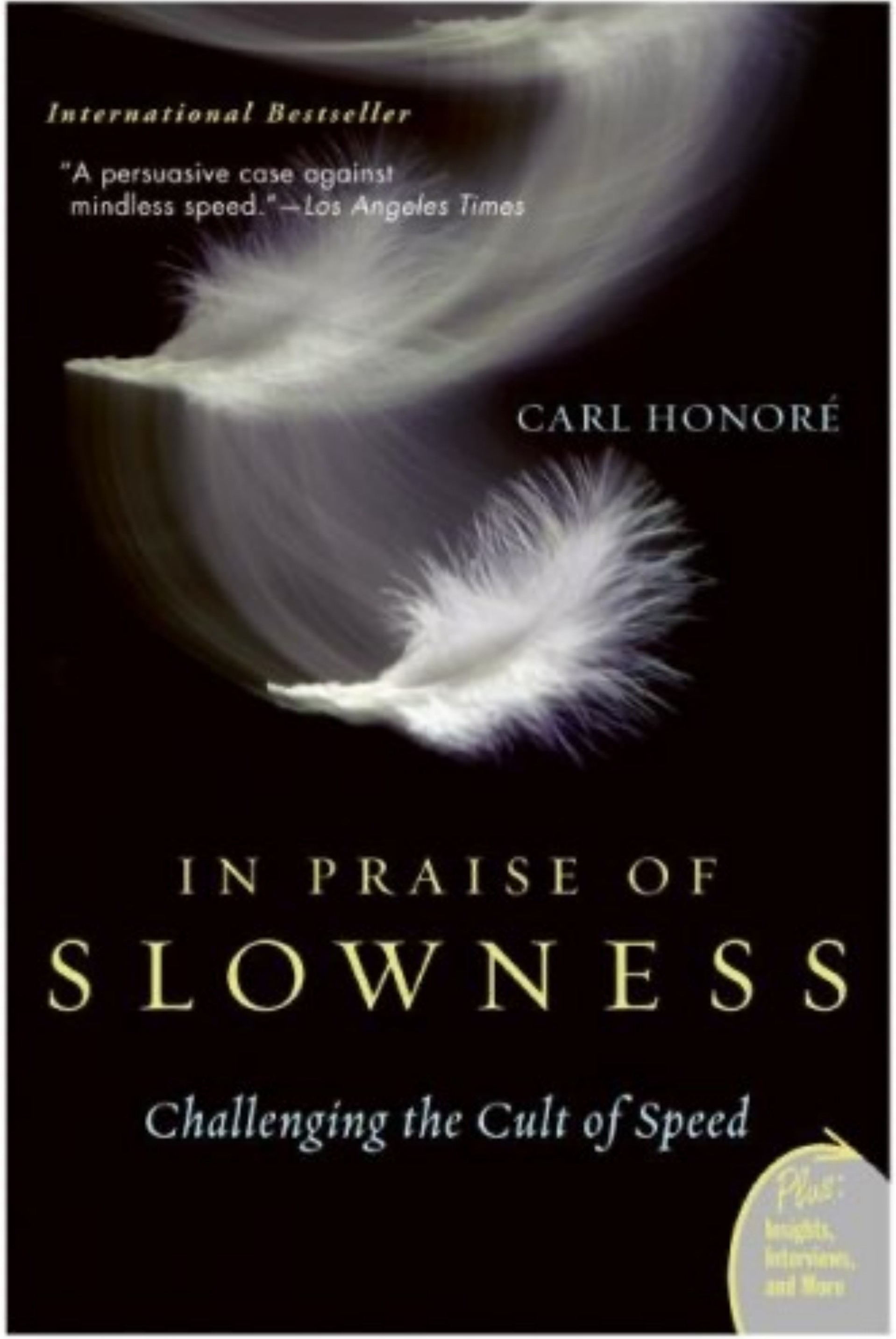
(slow (code) retreat)

MUREX™

@pbourgau
https://philippe.bourgau.net/
https://www.eventStormingJournal.com

Technical Agile Coach at Murex
https://www.murex.com/en/careers/overview
@Work_At_Murex

Why?


International Bestseller
"A persuasive case against mindless speed."—Los Angeles Times
CARL HONORÉ
IN PRAISE OF SLOWNESS
Challenging the Cult of Speed
Plus: Insights, Interviews, and More

The Slow Movement

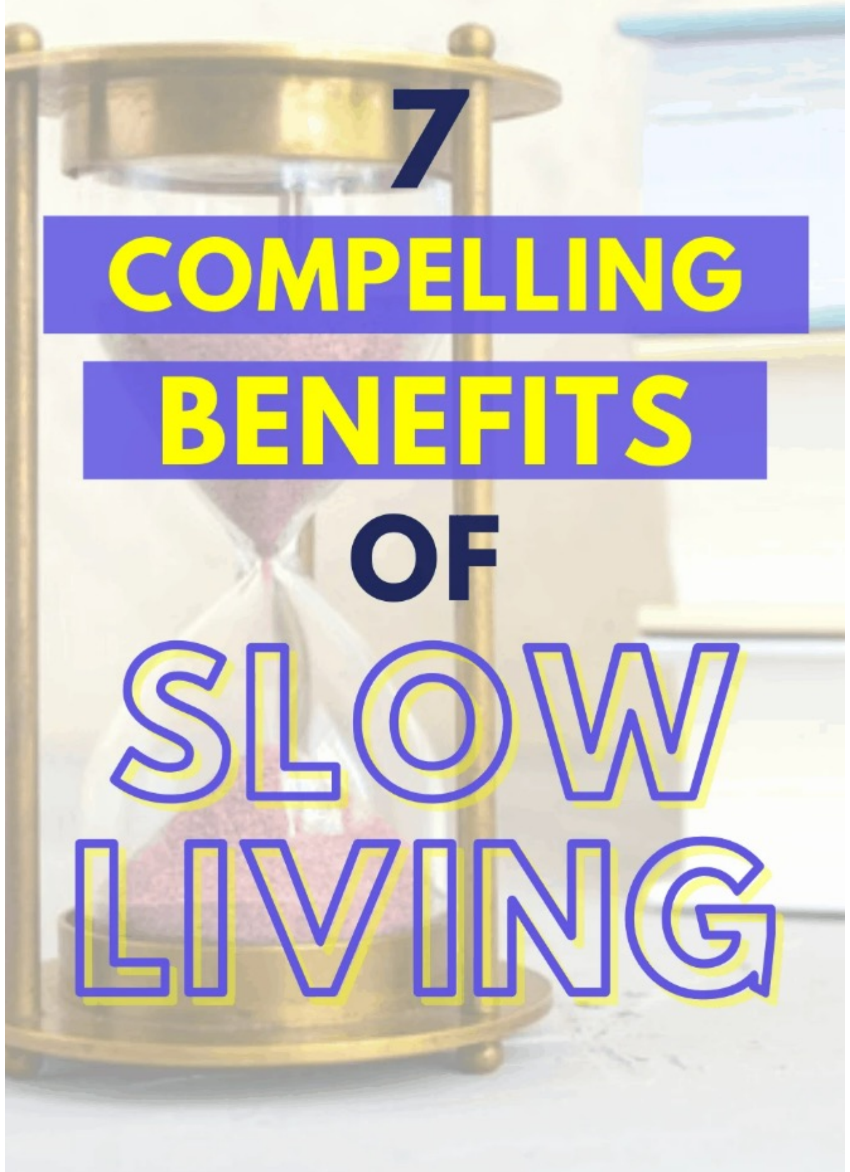
"It is a cultural revolution against the notion that faster is always better. The Slow philosophy is not about doing everything at a snail's pace. It's about seeking to do everything at the right speed. Savoring the hours and minutes rather than just counting them. Doing everything as well as possible, instead of as fast as possible. It's about quality over quantity in everything from work to food to parenting."

(From Carl Honoré's 2004 book, In Praise of Slow, cf Slow movement (culture) - Wikipedia)

Benefits of slow living

1. Drop the rat-race
2. Savor time
3. Focus on what matters most, like the people in your life
4. Can improve health
5. Richer and deeper experiences
6. Can boost your productivity and creativity
7. Lighter and happier life

(From [7 Compelling Reasons for Living a Slow and Simple Life - Fun Loving Families])



Benefits of Slow Work

1. More Focus
2. Improved Quality
3. Less Tiring
4. Less Multitasking
5. No Unnecessary Work

(From How Taking Things Slowly Can Help You Become More Productive (facilethings.com))

Why embrace slow work

- Working at a slower pace **prevents burnout**: the power of the process is that it can free up time and energy to do things that are not always reduced to a checklist.
- Slow work makes a **positive impact on sleeping and eating habits** for a more productive lifestyle.
- It **trains us to be internally calm** and more stable, thus, reducing reactivity and building a higher level of engagement and responsiveness.

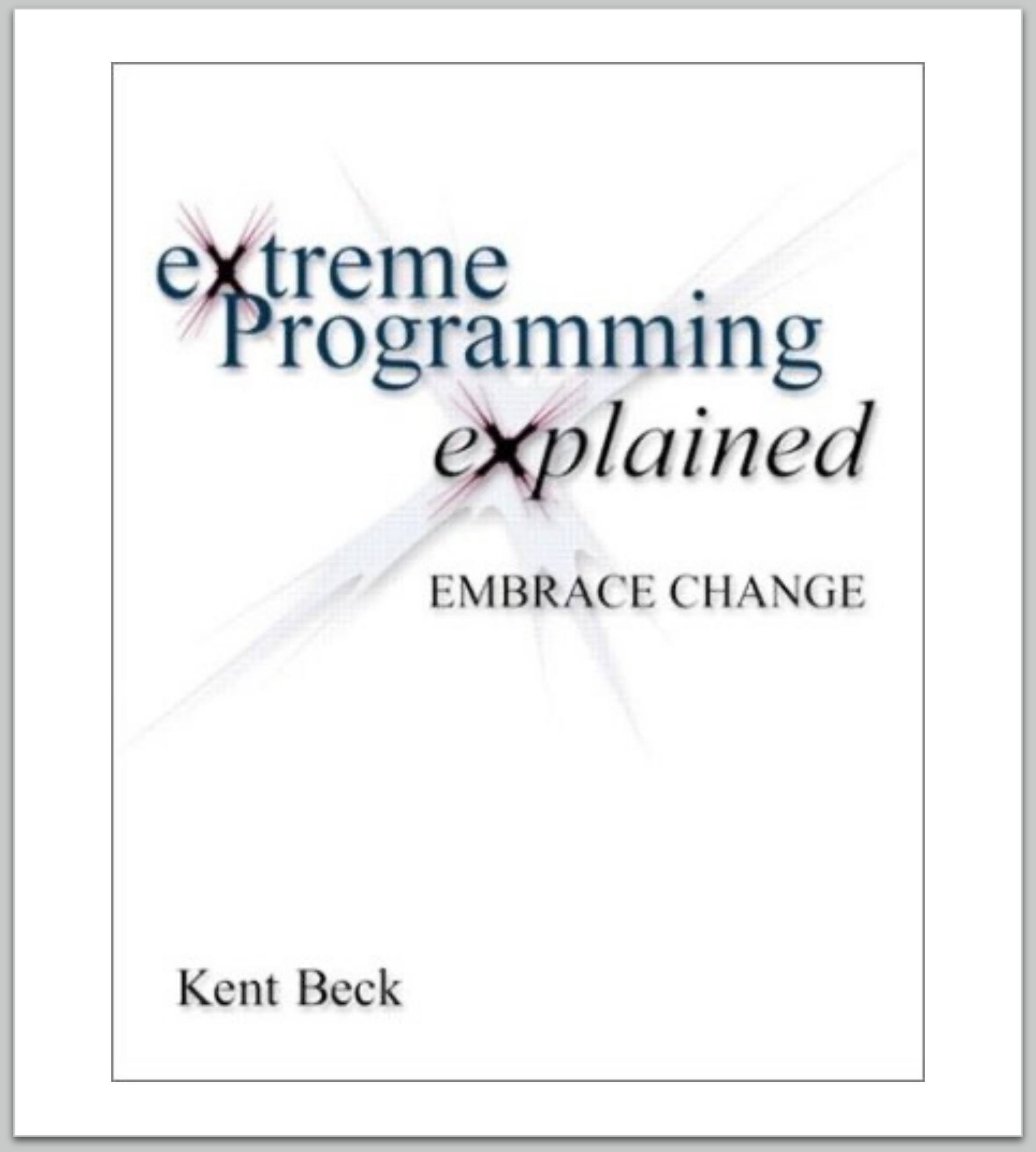
(From Why You Need Slow Work In 2022 (tmetric.com))

Slow Extreme Programming

*“XP is an experiment in answer to the question, "**How would you program if you had enough time?**" Now, you can't have extra time, because this is business after all, and we are certainly playing to win. But **if you had enough time, you would write tests; you would restructure the system when you learned something; you would talk a lot with fellow programmers and with the customer**.*

Such a "mentality of sufficiency" is humane, unlike the relentless drudgery of impossible, imposed deadlines that drives so much talent out of the business of programming. The mentality of sufficiency is also good business. It creates its own efficiencies, just as the mentality of scarcity creates its own waste.”

(From Kent Beck’s 1999 Book eXtreme Programming explained, Embrace Change)



Slow mobbing

Slow Driving

Take the time to be aware of your feelings and thoughts when driving:

- Accept to be navigated
- Stay focused on typing and do what you are asked to do
- Focus on the touch of the keyboard and what displays on the screen
- Don't judge what you are asked to do
- Breath while compiling or running tests

Benefits

- This driving 'pause' makes the overall mobbing more sustainable
- It also helps to take a step back, let your background creative mind work, and come back to navigation with a new perspective

References:

- [Mindfulness For Programmers. 3 simple exercises to be more present… | by Julia Di Russo | Towards Data Science]

Slow navigating

Take the time to practice egoless programming while navigating in the mob:

- Practice patience and compassion towards other mobsters when they seem to struggle with something you managed.
- Practice egoless programming and humility and welcome others’ solutions, especially when they are different from yours.
- Observe and let go of any self-criticism or self-judgment that you may notice when you struggle to understand or solve something.
- Variant: you can also take a step back in the mob. Strive to remain a neutral observer of the mob and keep a log of what it's happening, and what mood the mob is going through.

Benefits:

- A compassionate behavior builds psychological safety in the team, which improves decision making and overall team performance.
- A mob log can be very useful for continuous improvement and retrospectives.

Slow Communication

Slow code reviews

Take the time to offer non-violent feedback to your colleagues:

- When writing reviews
- When voicing feedback in the mob
- Follow the Non-Violent Communication pattern:
  - List the facts
  - Share your feelings
  - Share what needs you have
  - Suggest a new way of doing things that meets everyone's needs

Benefits:

- Avoids conflicts to build-up for the wrong reasons
- This improves teamwork, psychological safety, but also the efficiency of reviews
- Leaves the door open for more creative solutions
- Build our feeling and need fluency and self-awareness
- A way to start learning Non-Violent Communication

References:

- [code-in-flow/mindful-programming (github.com)]()

Slow Coding

Slow self-retros

Take the time to observe what happens inside us:

- Set a timer to ring every 5 minutes
- Every time it rings, take a short pause and fill out this log individually:

ROTI	Did	Learnt	Puzzles	Feelings	Needs	Decisions

- You can use cheat sheets from Non-Violent Communication for ideas for feelings and needs

Benefits

- Helps us to spot our non-productive reactions to events like:
  - “I don’t know how to do that!”
  - “Damn I was sure this test would pass! I’ve got no idea why it’s failing!”
  - “How stupid I am!”
  - “I don’t understand anything about this code!”
- Develops our feeling and needs fluency
- Provides plenty of data for continuous improvement and retrospectives of all kinds

References:

- code-in-flow/mindful-programming (github.com)

Slow steps

Take the time to practice egoless programming, baby-steps programming, and present focus with TCR:

- Start coding with TCR
- Whenever you get reverted
  - notice the feelings you experience and accept that err is humane.
  - Practice humility
  - Try again with a smaller step
- As the day goes by, notice how your capacity for flawless work decreases!

Benefits

- With time you'll learn how much you can do without mistakes
- This teaches you to do extra small steps, that cannot go wrong!
- This helps to understand when to call it a day and to leave
- This avoids writing low quality code when you are too tired

Reference:

- code-in-flow/mindful-programming (github.com)
- 2 years of TCR pres by Xavier Detant at Alpes Craft 2022

Slow TODO lists

Take the time to focus only on the present task:

- Keep a TODO list while you code
- Keep track of everything that remains to be done
- Check or strikethrough tasks as you do them
- Indent sub-tasks
- Can use a .txt, .markdown, a shared online doc, or a mind map
- Reorganize your tasks when needed
- If you revert what you cannot do yet, it becomes the Mikado Method

Benefits:

- This removes all the mental load involved with keeping track of the status of work
- The map serves as a communication tool for handovers, pairing, or mobbing
- Helps us to drop things that we eventually decide, not to do!

Reference:

- code-in-flow/mindful-programming (github.com)
- TO DO list or Mind Map for programming

Useful Downloads

Get the Code

https://github.com/philou/Kata-Slow-Code

(Java with TCR)

https://cyber-dojo.org/

(More languages, no TCR)

Get the Slides

https://1drv.ms/b/s!Auooa9luG_aGo0ktZI47hychBGc4?e=8QjeGQ

Setup steps

Local repo with Java and TCR

1. Clone the repo https://github.com/philou/Kata-Slow-Code
2. Open the project file in your IDE (build.gradle or pom.xml)
3. Make sure the tests run
4. Open a git-bash terminal (should work from IDE)
5. Go to the java sub-directory
6. Run `./trcw solo`

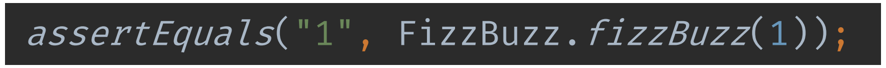
Cyber Dojo, any language, no TCR

1. Go to https://cyber-dojo.org/
2. Create a new session of Fizz Buzz in your preferred language
3. Make sure the tests run

Cheat sheet

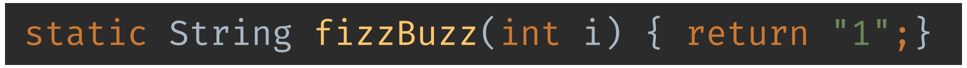
TDD

1. Write a failing test



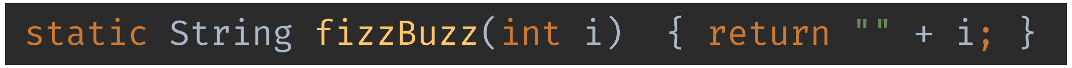
```
assertEquals("1", FizzBuzz.fizzBuzz(1));
```

2. Pass the test, easily and quickly



```
static String fizzBuzz(int i) { return "1";}
```

3. Refactor the code



```
static String fizzBuzz(int i)  { return "" + i; }
```

Strong Style mobbing

- The driver just types what he is asked to
- The navigator tells the driver what to do
- Round rob roles every 5 minutes

FAQ

How to apply it in day-to-day programming?

- Some practice don't take time and will actually make you faster! Slow TODO list will always make you more efficient! The mob is a great place to start slow-code, since the mob moves on whatever you do. You can already practice slow driving, slow navigating, slow self-retros, and slow reviews in the mob.
- Other will take a bit of practice to get better. Start 30 minutes daily until you manage to get it better. It should be a nice moment, I also found it a way to reconnect to the joy of programming. This should help you to distill a bit of 'slow' in the rest of your day. You might also find some colleagues to practice slow-mobbing every day for a half-an hour or so. Like TDD slow-code requires some practice, but eventually, it's more effective.
- You can also start slow self-retros by pausing every 15 minutes or so, and little by little increase the frequency as you get better at it.
- If you don't have the time to do proper slow-review, this might be an incentive to do more (slow-) pairing and mobbing!

What about other non-slow aspects of our work (meetings, interruptions).

- I think there is some literature on the web about that. I did not dig yet, but I'm sure there is something to hack!

This is non-sense! How can we go slow, the whole industry is about going "Fast"!

- I have no silver-bullet, I just have the conviction that, even inside a whirlwind of activity, practicing the ability to go slow is an opportunity to avoid stress and to have more impact.

Hand notes

Feelings Help Sheet

If you need, you can use this list of feelings to help you fill out the self-retro ‘feelings’ column.
Keep it short and easy, don’t hesitate to stick to categories.

- Fear
  - Rejected
  - Submissive
  - Insecure
  - Anxious
  - Scared
  - Worried
- Surprise
  - Startled
  - Confused
  - Amazed
  - Excited
- Happiness
  - Joyful
  - Interested
  - Proud
  - Accepted
  - Powerful
  - Peaceful
  - Intimate
  - Optimistic
  - Arrogant
  - Appreciative
- Anger
  - Hurt
  - Threatened
  - Hateful
  - Mad
  - Aggressive
  - Frustrated
  - Distant
  - Critical
  - Indignant
  - Offended
- Disgust
  - Disapproval
  - Disappointed
  - Awful
  - Avoidance
- Sadness
  - Guilty
  - Abandoned
  - Despair
  - Depressed
  - Lonely
  - Bored

Inspired from [Emotional Cheat Sheet (updated from emotional wheel chart) : writing (reddit.com)]

Human Needs Help Sheet

If you need, you can use this list of needs to help you fill out the self-retro ‘needs’ column.
Keep it short and easy, don’t hesitate to stick to categories.

- Self actualization
  - Personal Growth
  - Fulfilment
  - Inspiration
  - Harmony / Peace
- Physical Nurturance
  - Food
  - Water
  - Shelter
  - Sleep Reset
  - Sexual Expression
  - Physical Activity / Movement
  - Air / Oxygen
  - Touch
  - Warmth
- Meaning
  - Achievement
  - Autonomy
  - Responsibility
  - Celebration
  - Needed
  - To understand and be understood
  - To see and be seen
  - Learning
  - Self-expression
  - Challenge
  - Clarity
  - Hope
  - Understanding
- Belonging and love
  - Support
  - Affection
  - Relationships
  - Work
  - Appreciation
  - Community
  - Communication
  - Attachment
  - Intimacy
  - Connection
  - Respect / Self-respect
  - Trust
  - Acceptance
  - Closeness
- Safety
  - Protection
  - Stability
  - Law / Order
  - Empowering Honesty
  - Security
  - Limits / Boundaries
- Integrity
  - Authenticity
  - Creativity
  - Self-Worth
- Play
  - Joy
  - Fun
  - Laughter

Source https://www.habitsforwellbeing.com/basic-human-needs-are-your-needs-being-met/

Take 1’ to self-retrospect every 5’

ROTI	Did	Learnt	Puzzles	Feelings	Needs	Decision

Take 1’ to self-retrospect every 5’

ROTI	Did	Learnt	Puzzles	Feelings	Needs	Decision