Experience Hero Tech Course Pre-Course

20222191 puleugo

Why are we doing this assignment?

- Most of us are probably just doing it because we’re told to.

Why are we doing this assignment?

- ~~Most of us are probably just doing it because we're told to.~~
- But ultimately, it's to become developers who build good software.

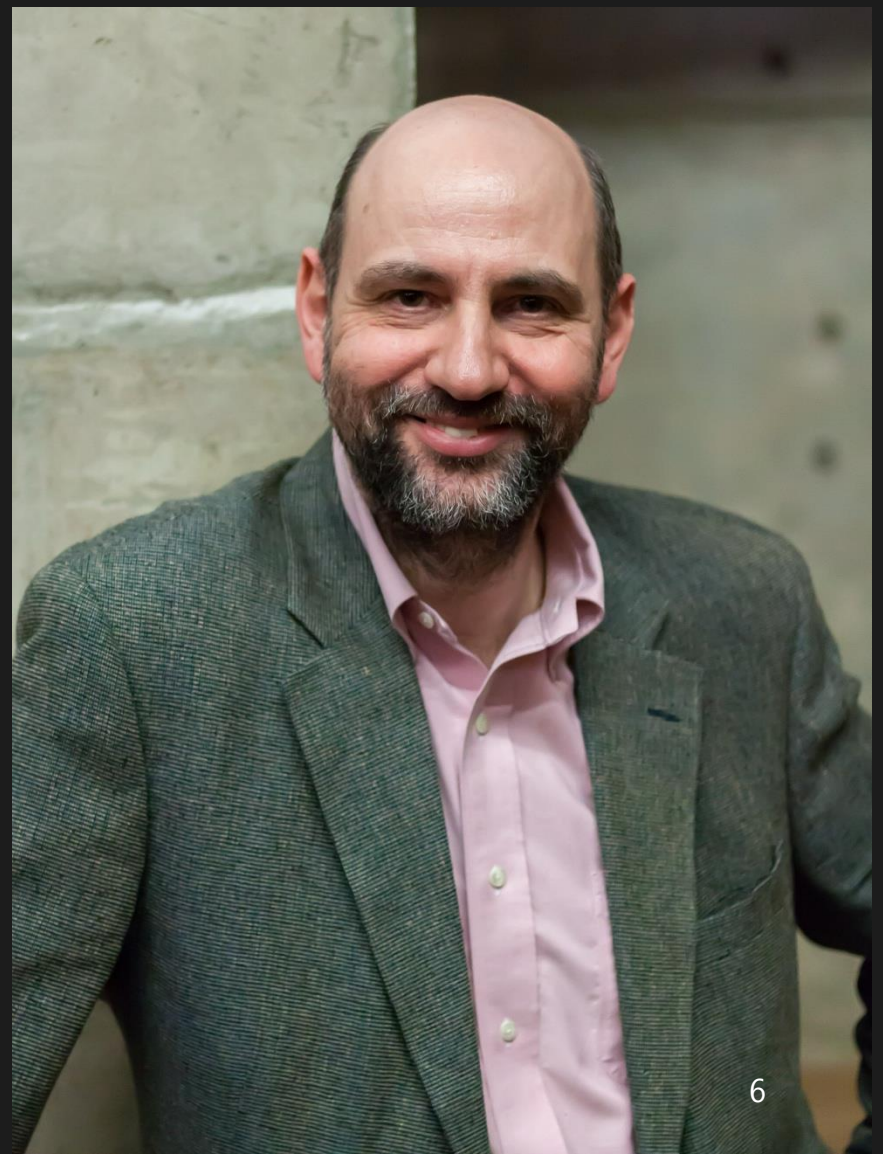
Why are we doing this assignment?

- ~~Most of us are probably just doing it because we're told to.~~
- But ultimately, it's to become developers who build good software.

Why are we doing this assignment?

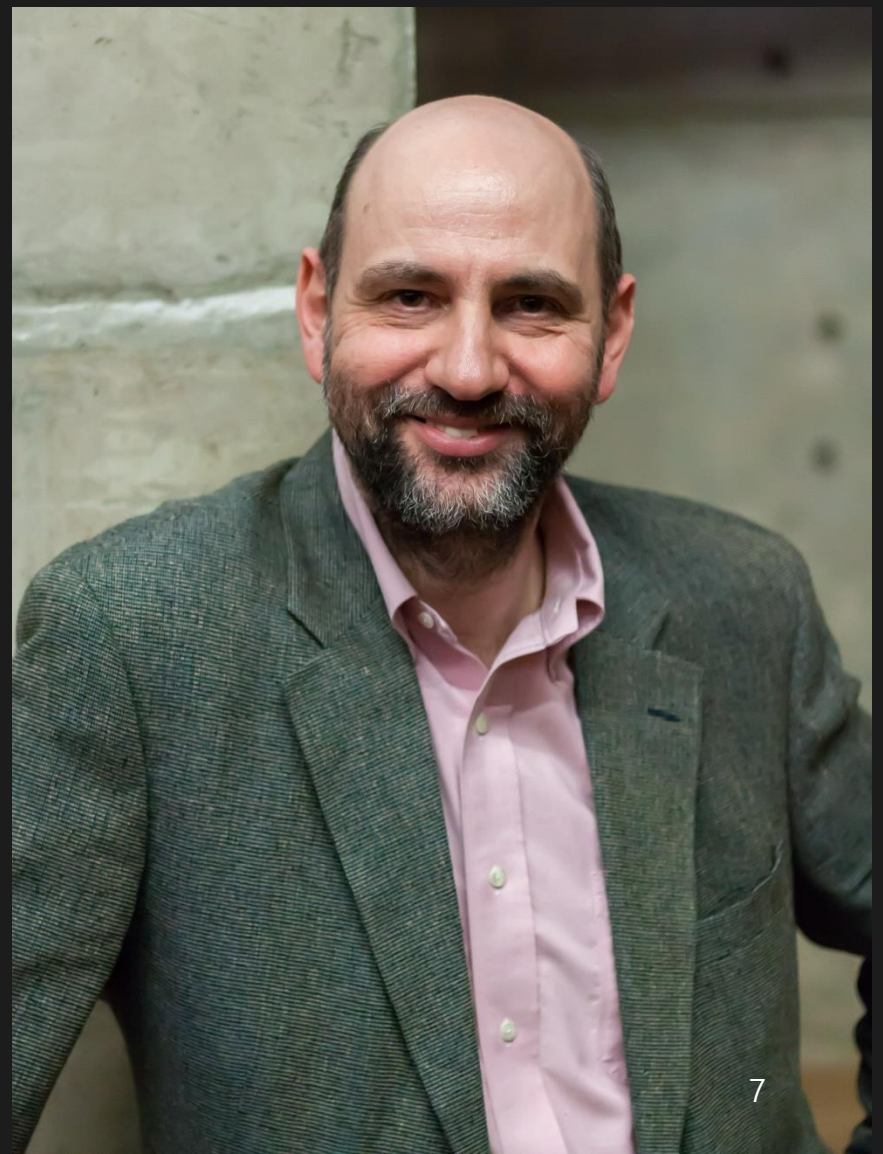
- ~~Most of us are probably just doing it because we're told to.~~
- But ultimately, it's to become developers who build good software.
  - So what is "good software"?

What is good software design?

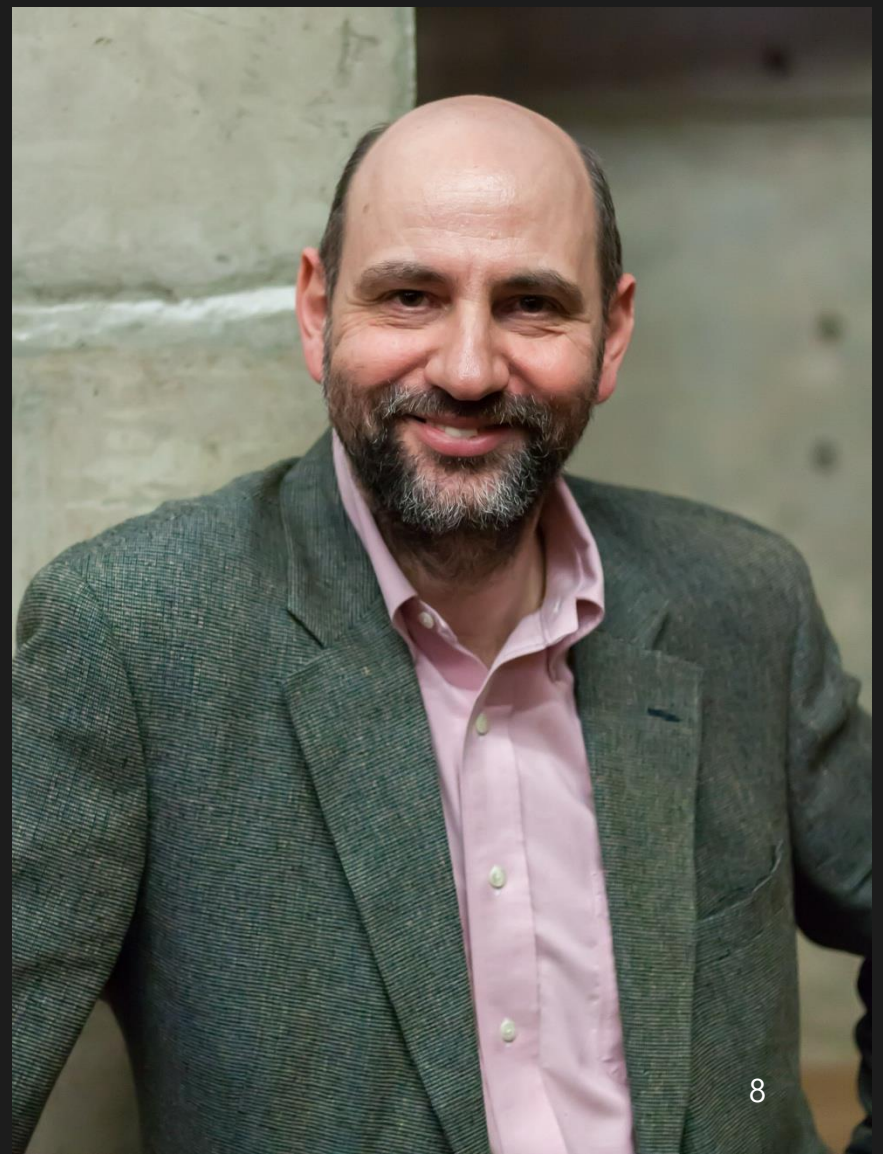


What is good software design?

- TODO:
  Write the number and names of classes used in your RacingCar implementation in the group chat.

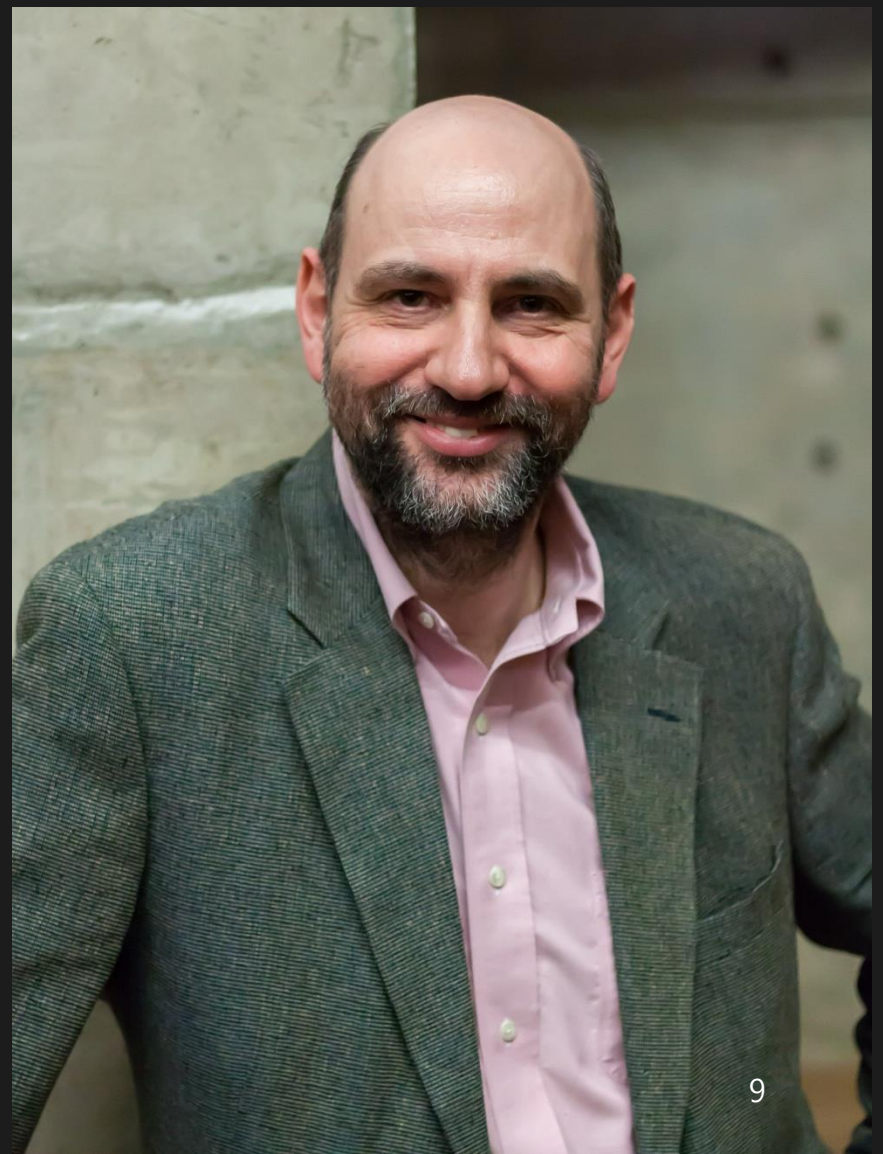


Lessons from 45 years of experience



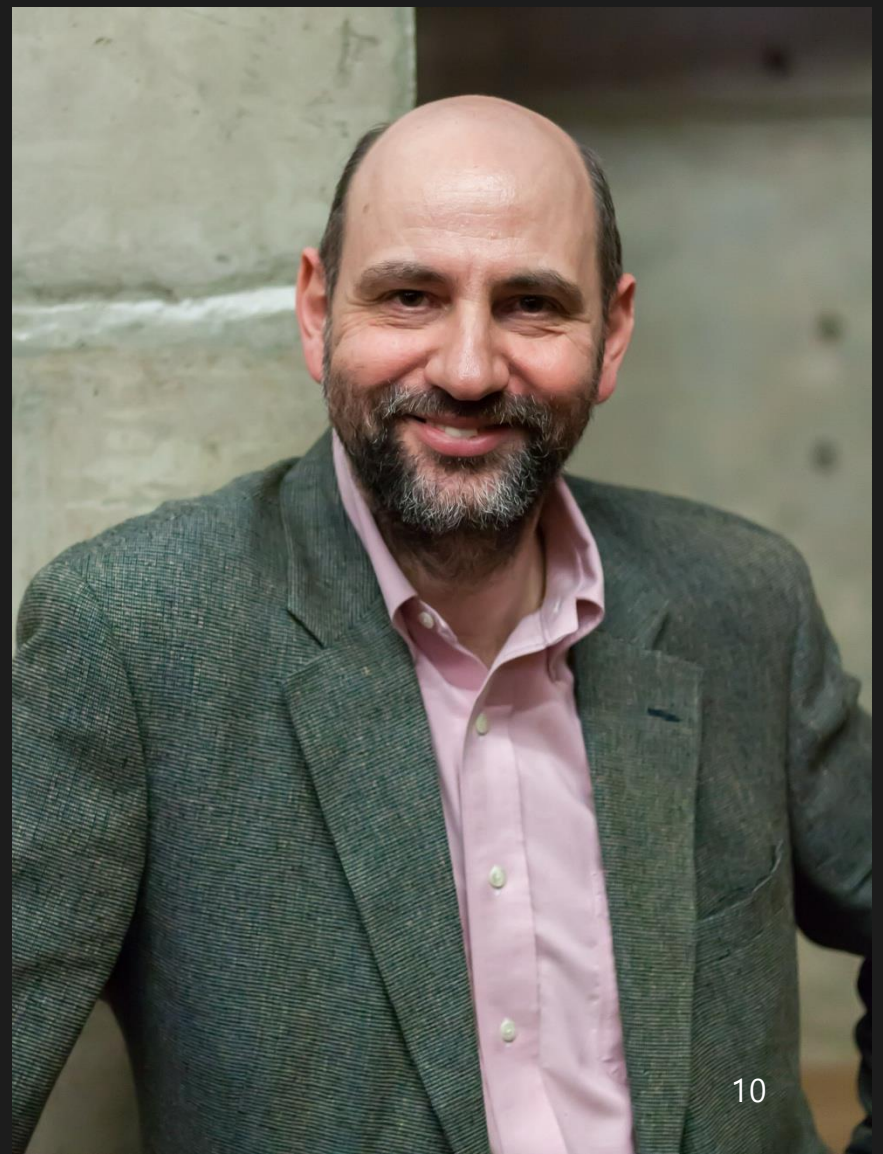
Lessons from 45 years of experience

1. The customer doesn't care **at all** about your code.
2. The customer is a pain.



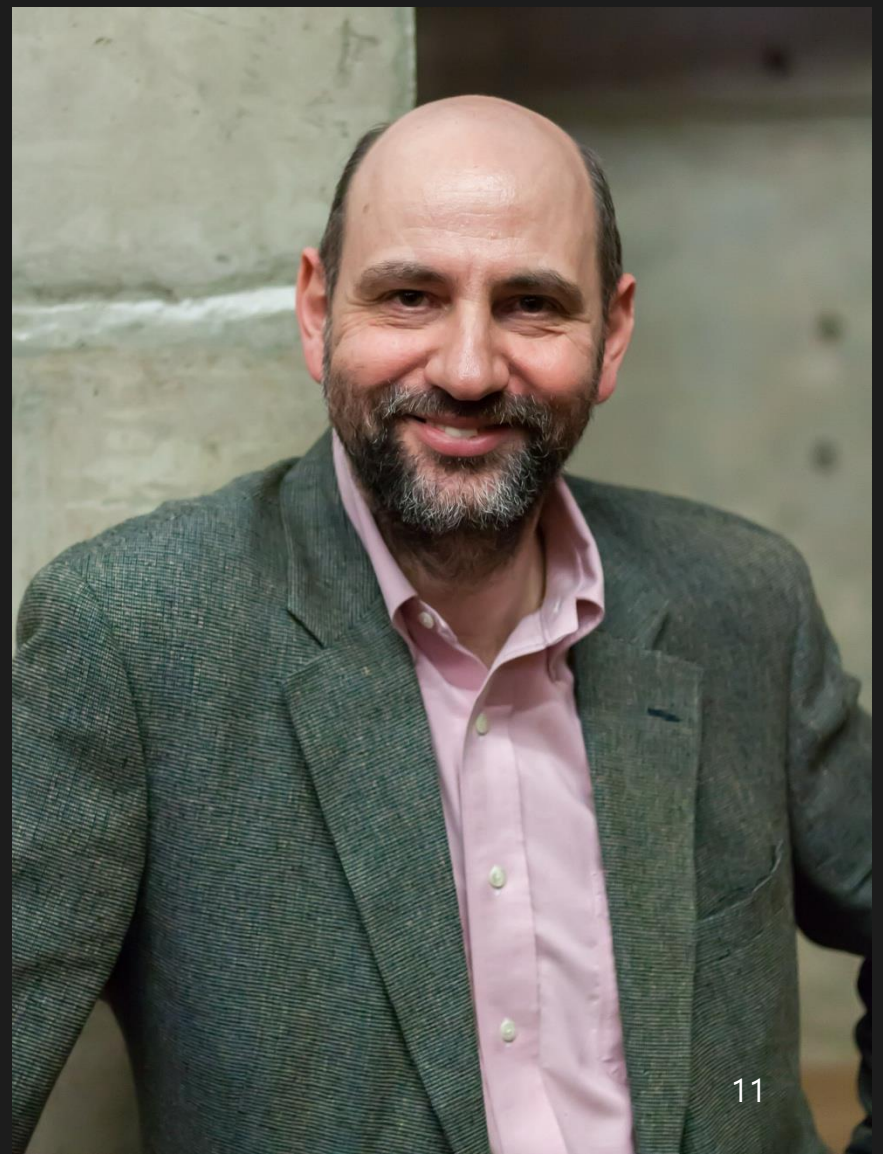
Lessons from 45 years of experience

1. The customer doesn’t care **at all** about your code.
   - If they ask you to explain your code a month later and you freeze? You're dead.
2. The customer is a pain.



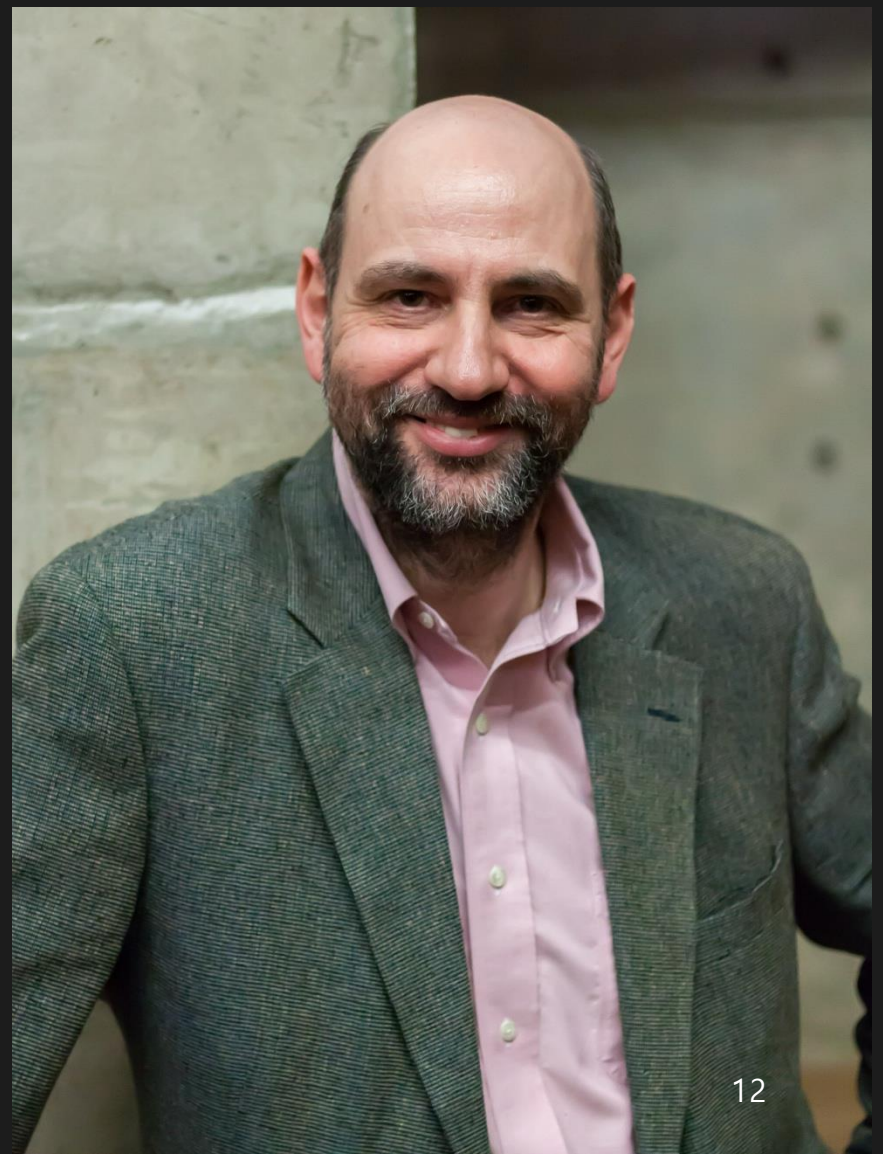
Lessons from 45 years of experience

1. The customer doesn’t care **at all** about your code.
   - If they ask you to explain your code a month later and you freeze? You're dead.
   - So they’ll just pressure the developer.
2. The customer is a pain.



Lessons from 45 years of experience

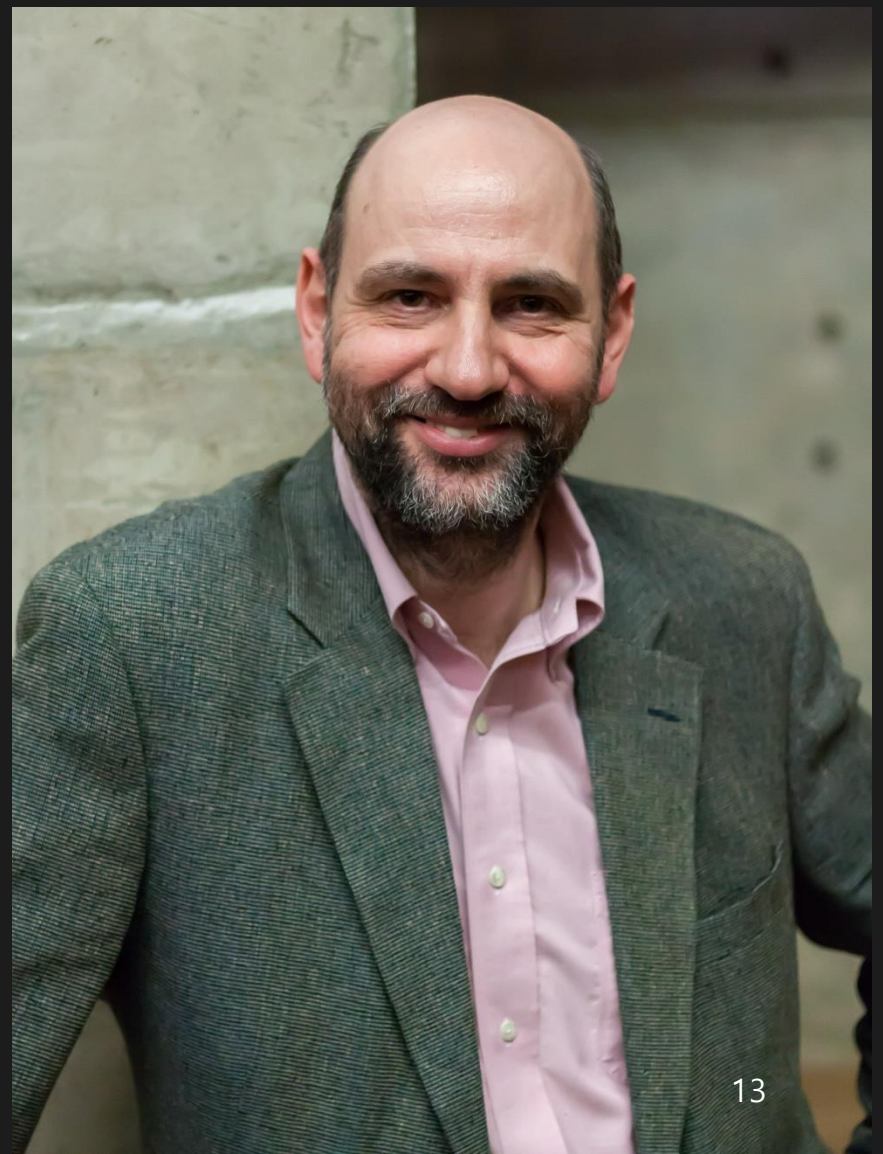
1. The customer doesn’t care **at all** about your code.
   - If they ask you to explain your code a month later and you freeze? You're dead.
   - So they’ll just pressure the developer.
2. The customer is a pain.
   - They change requirements however they want.



Lessons from 45 years of experience

1. The customer doesn’t care **at all** about your code.
   - If they ask you to explain your code a month later and you freeze? You're dead.
   - So they’ll just pressure the developer.
2. The customer is a pain.
   - They change requirements however they want.

   → But we can’t pressure the customer.

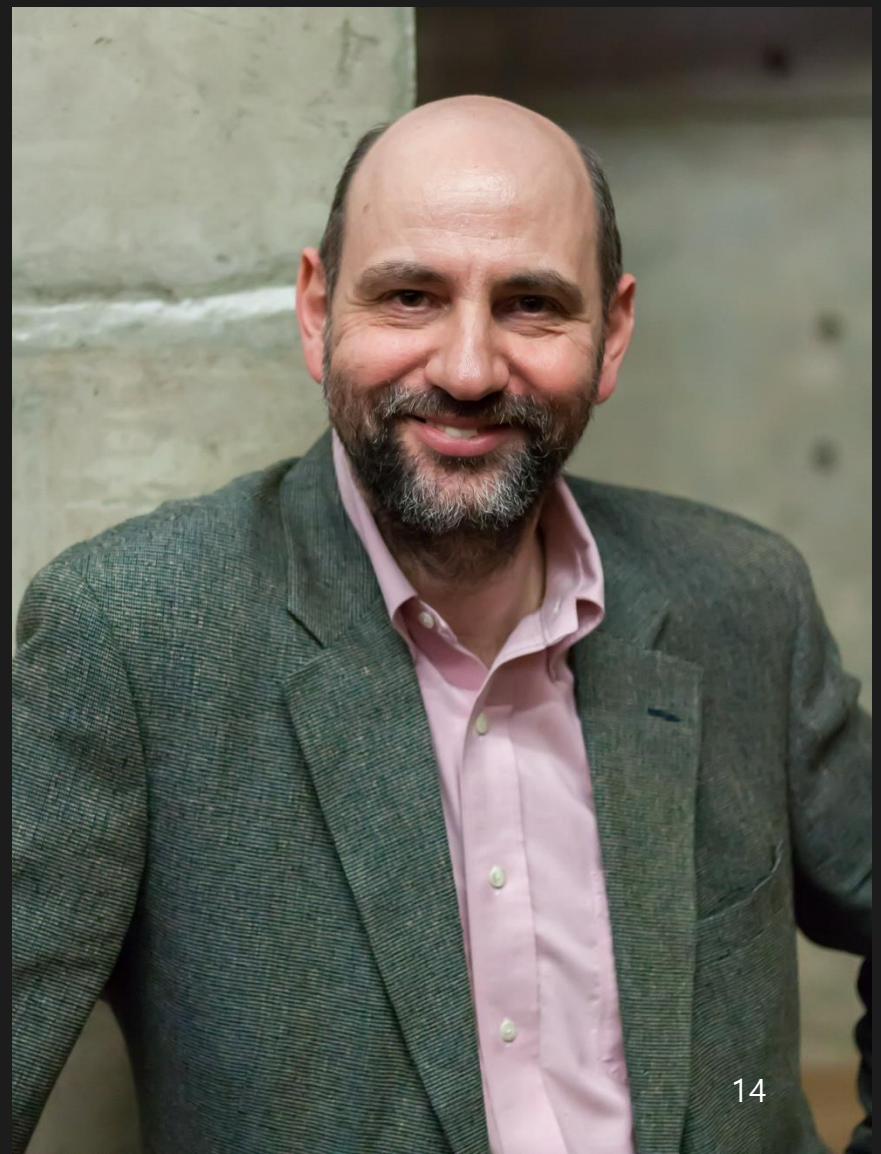


Lessons from 45 years of experience

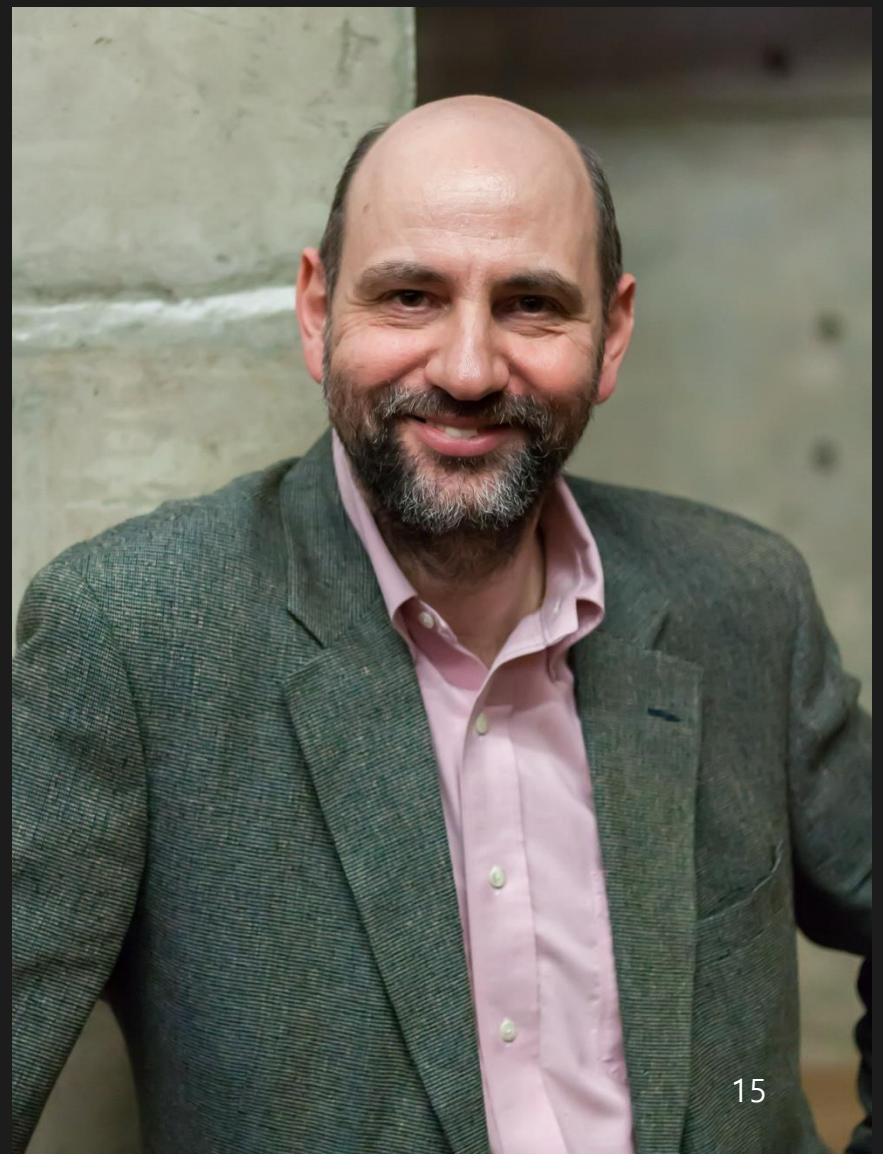
1. The customer doesn't care **at all** about your code.
   - If they ask you to explain your code a month later and you freeze? You're dead.
   - So they'll just pressure the developer.
2. The customer is a pain.
   - They change requirements however they want.

   → But we can't pressure the customer.

   → So write code considering **expandability**.

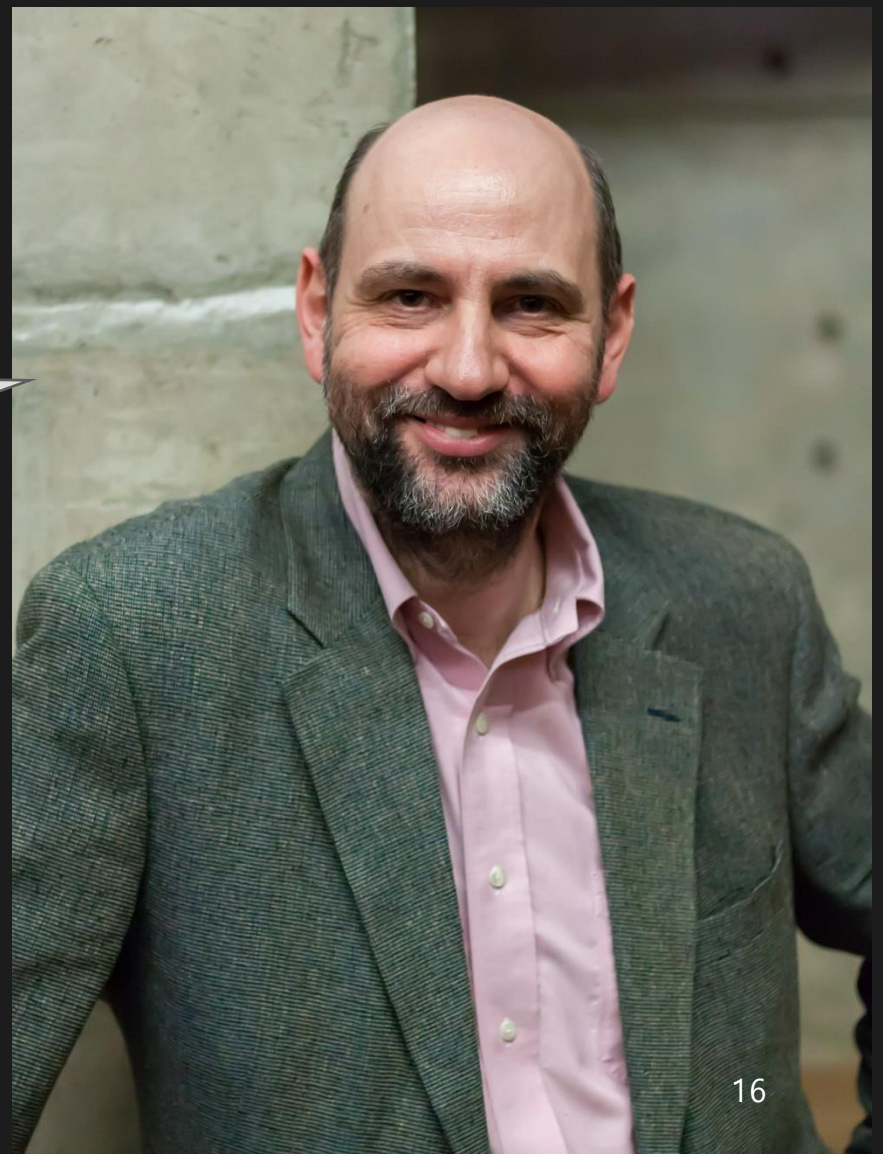


So What is good Software?



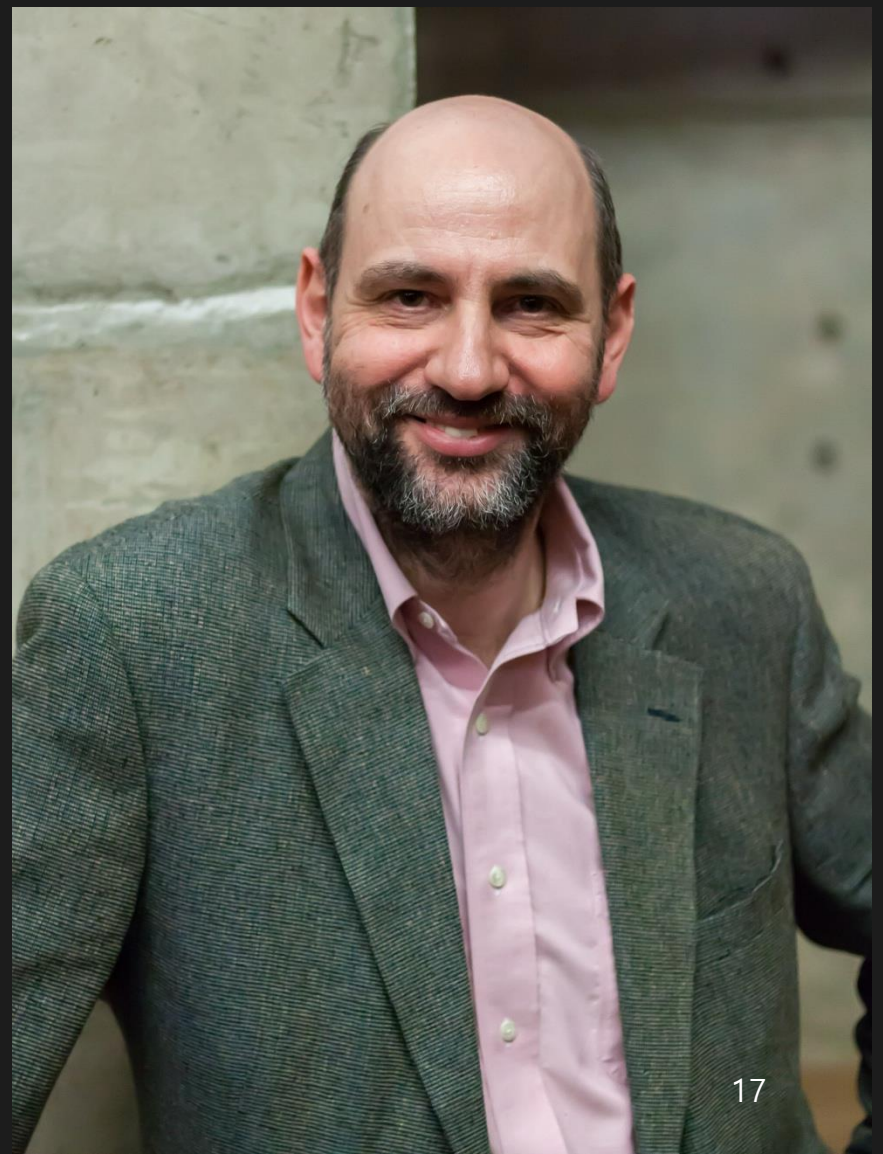
So What is good software design

Being **open to extension** is enough.



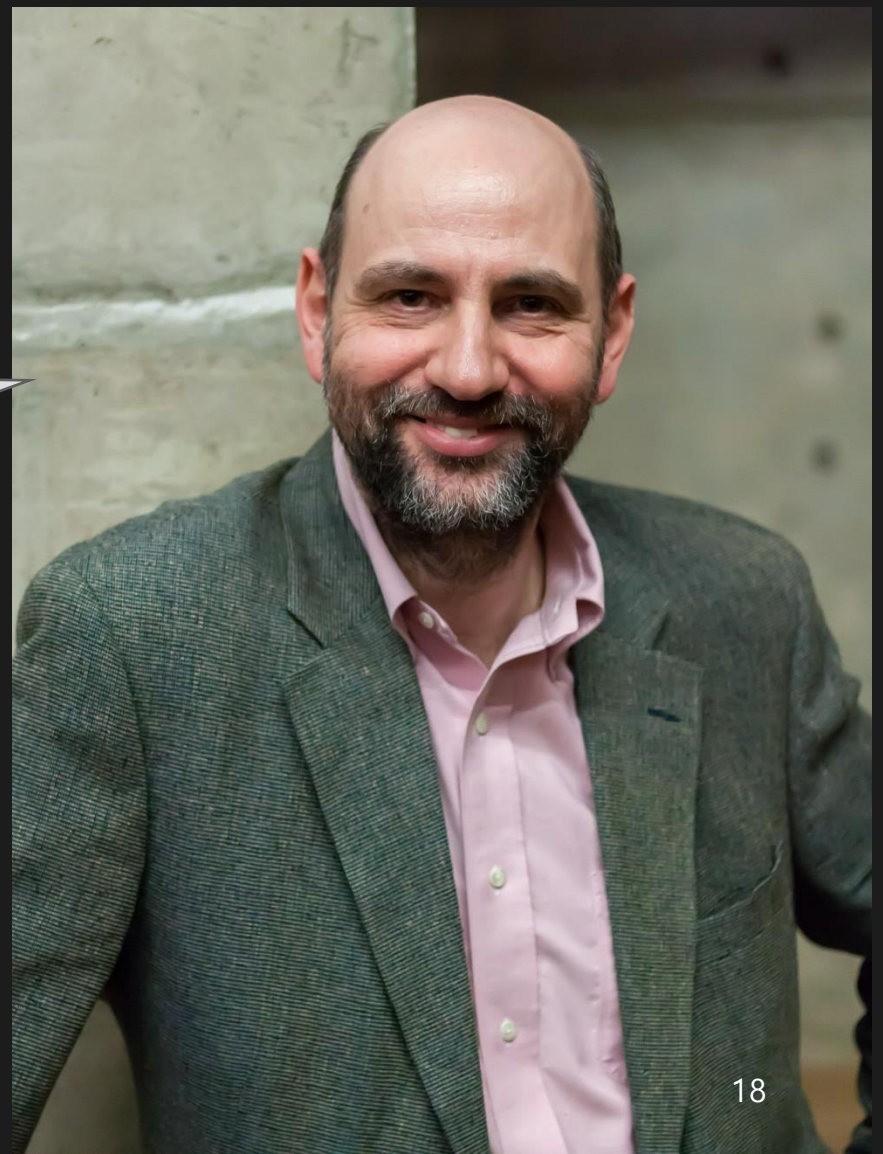
So What is good software design?

- S.O.L.I.D principles?
- Design patterns?



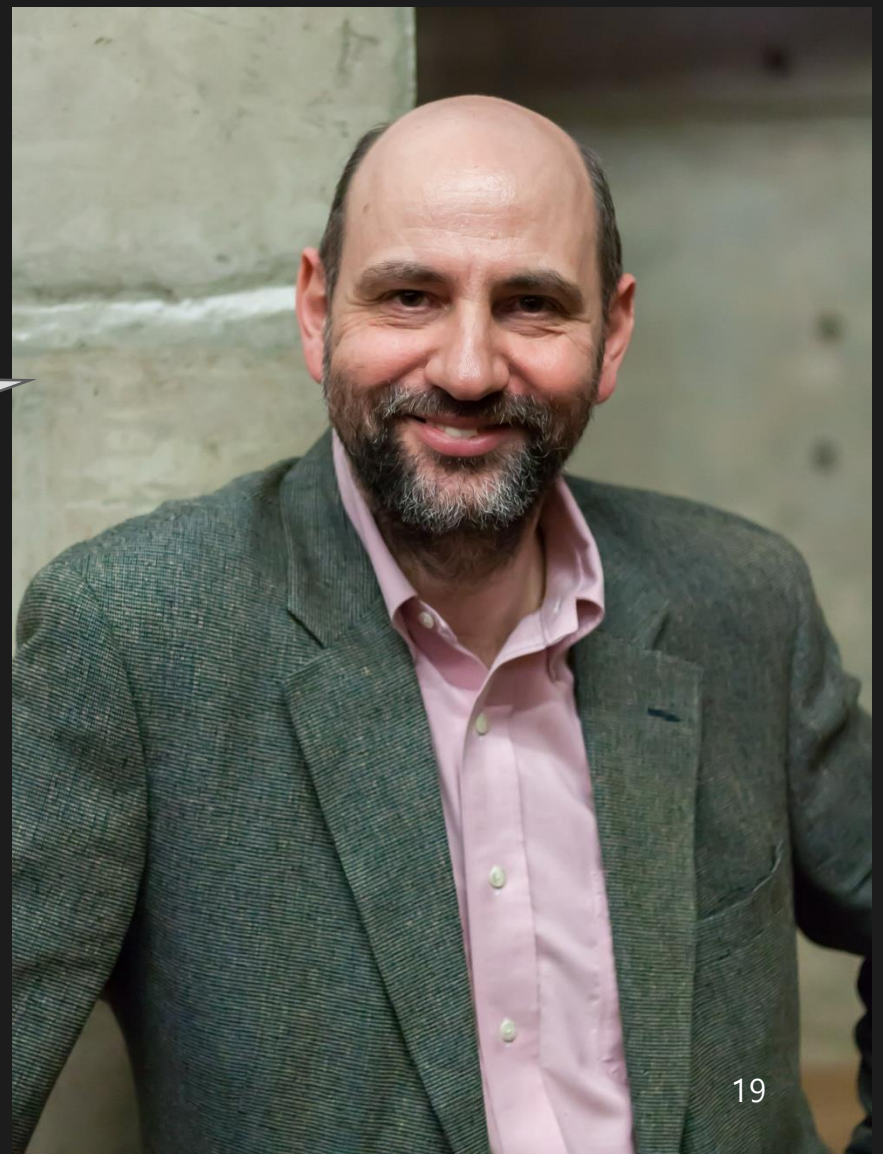
Fact

Honestly, you wouldn't understand even if I explained.



Fact

You just have to get beaten up (by experience).



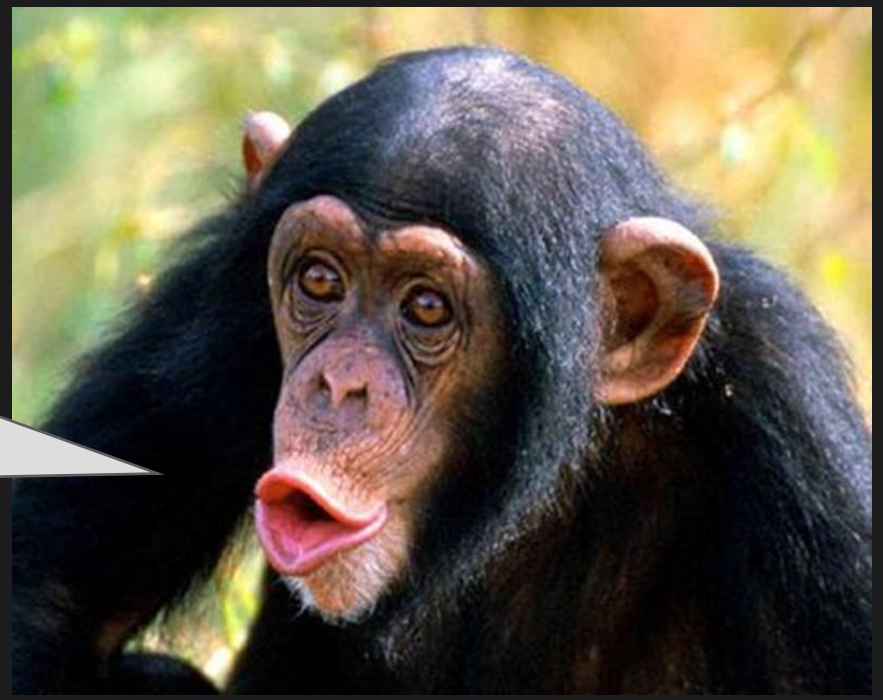
A taste of annoying customers

A taste of annoying customers

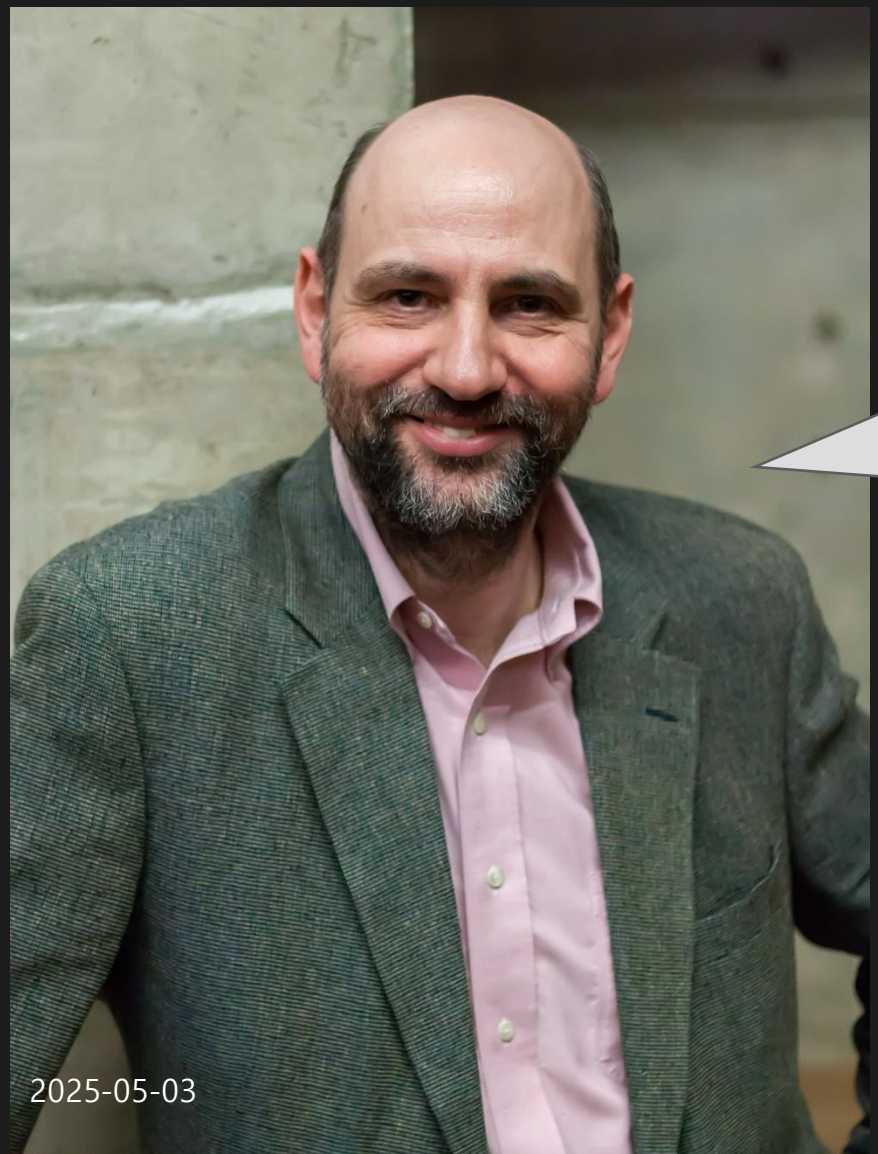
- Green class names: If you implemented them, your separation of concerns is probably good (expandable).
- Red class names: Classes added for new requirements. It's normal if they're not there yet.
- Just matching a few green classes doesn’t measure design skill—it’s just a reference.

A taste of annoying customers

Make each car have different movement conditions.



Make each car have different movement conditions.



OK, Inject move condition in the *Car constructor.

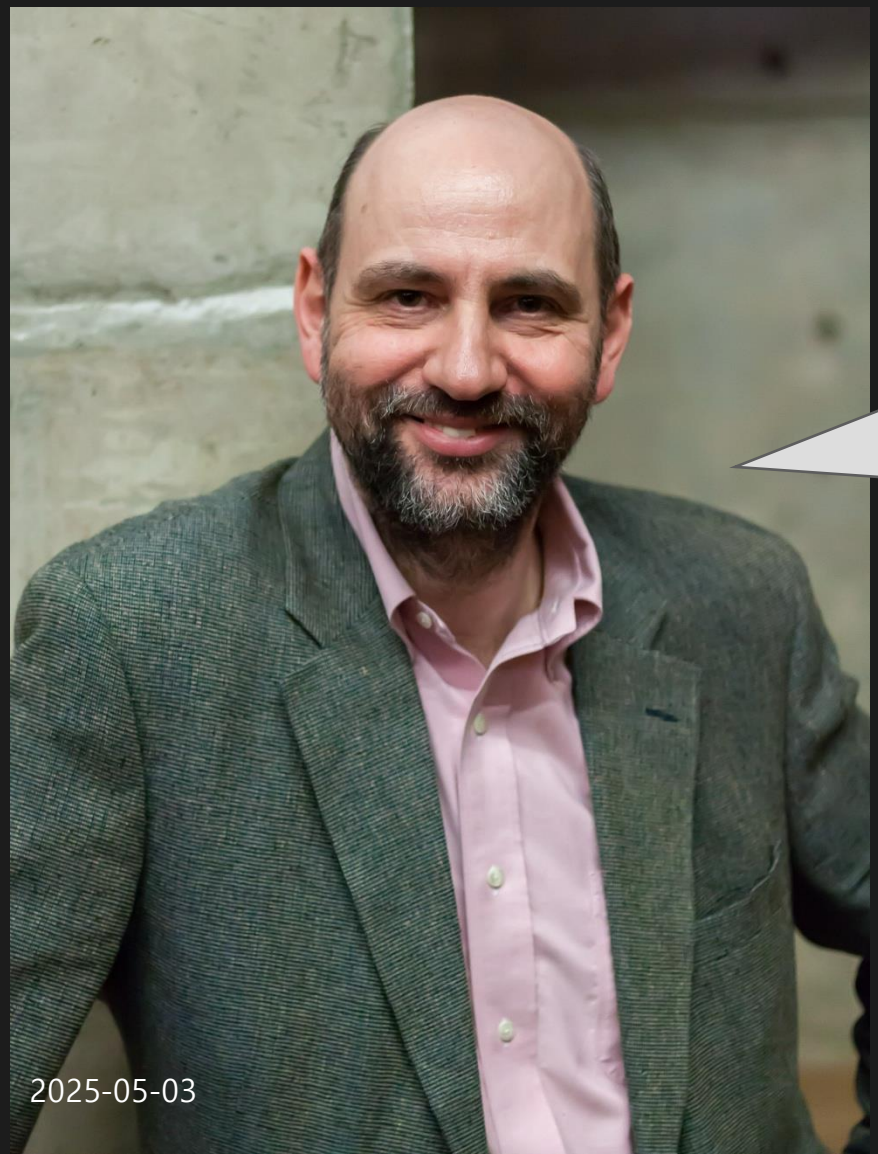
Code Change Lines: 1

- Car: Domain Model

A taste of annoying customers

No, let them be randomly assigned.

Make each car have different movement conditions.



Just randomize the
*moveCondition field
read by *isMovable().

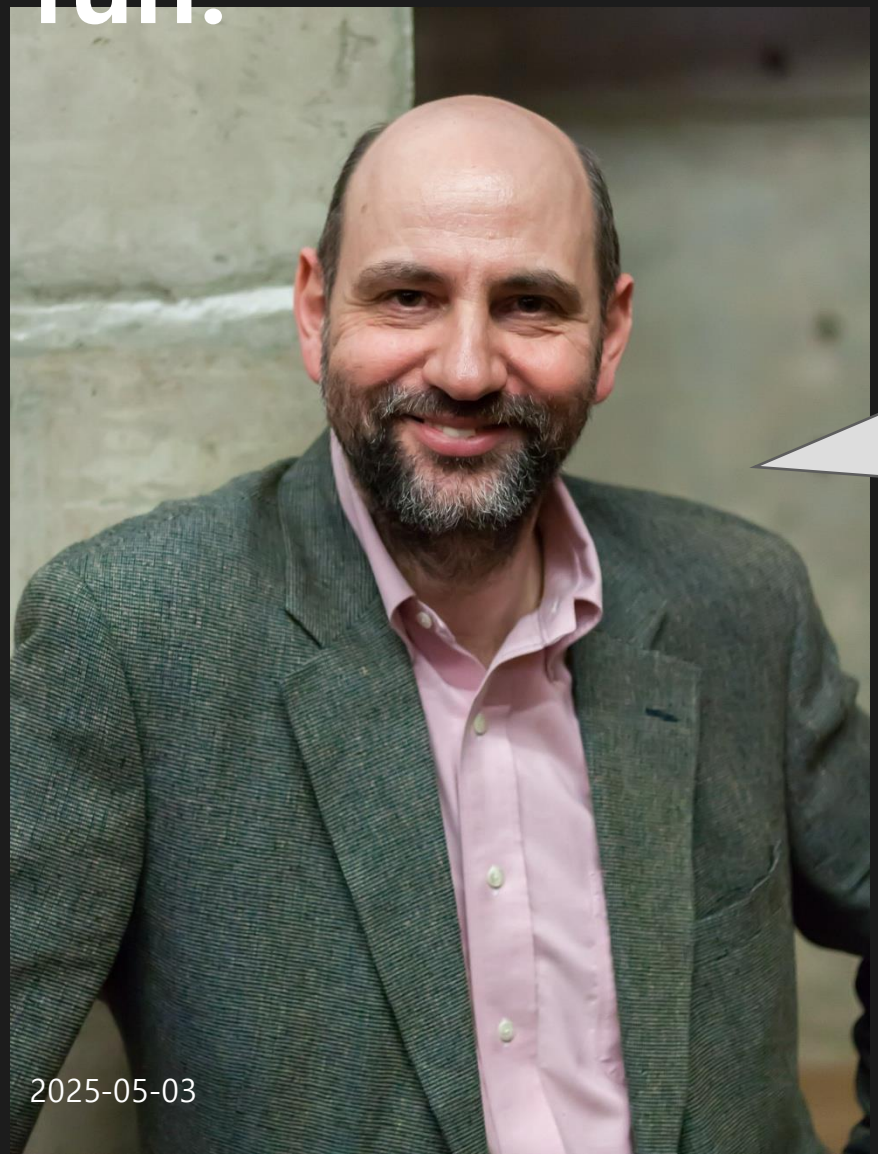
Code Change Lines: 1

- Separate functions so that each one does only a single task.
- If a value doesn't disappear after a single use, promote it to a field variable.

A taste of annoying customers

Oh, I think the game would be more fun... Please add obstacles.

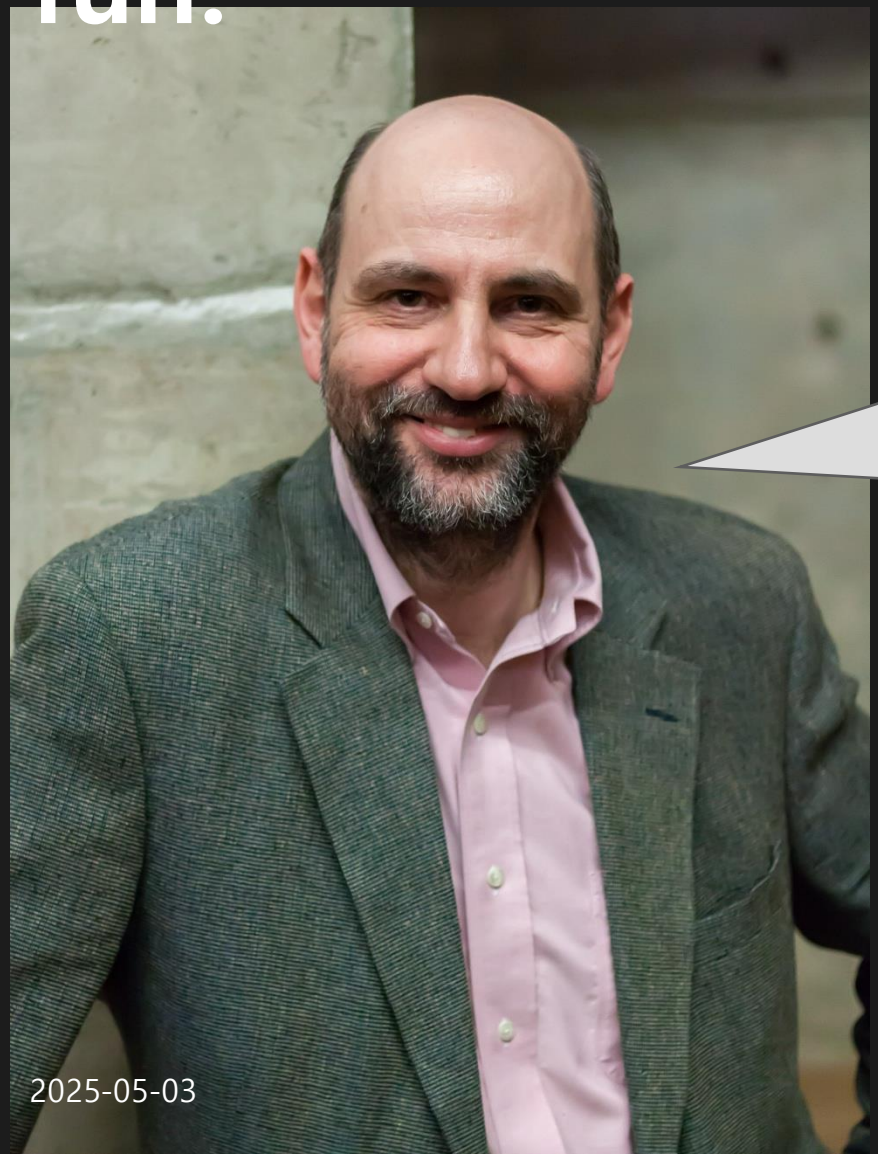
Add obstacles to make the game more fun.



Alright, in the Round class, I'll spawn obstacles every 5 rounds.

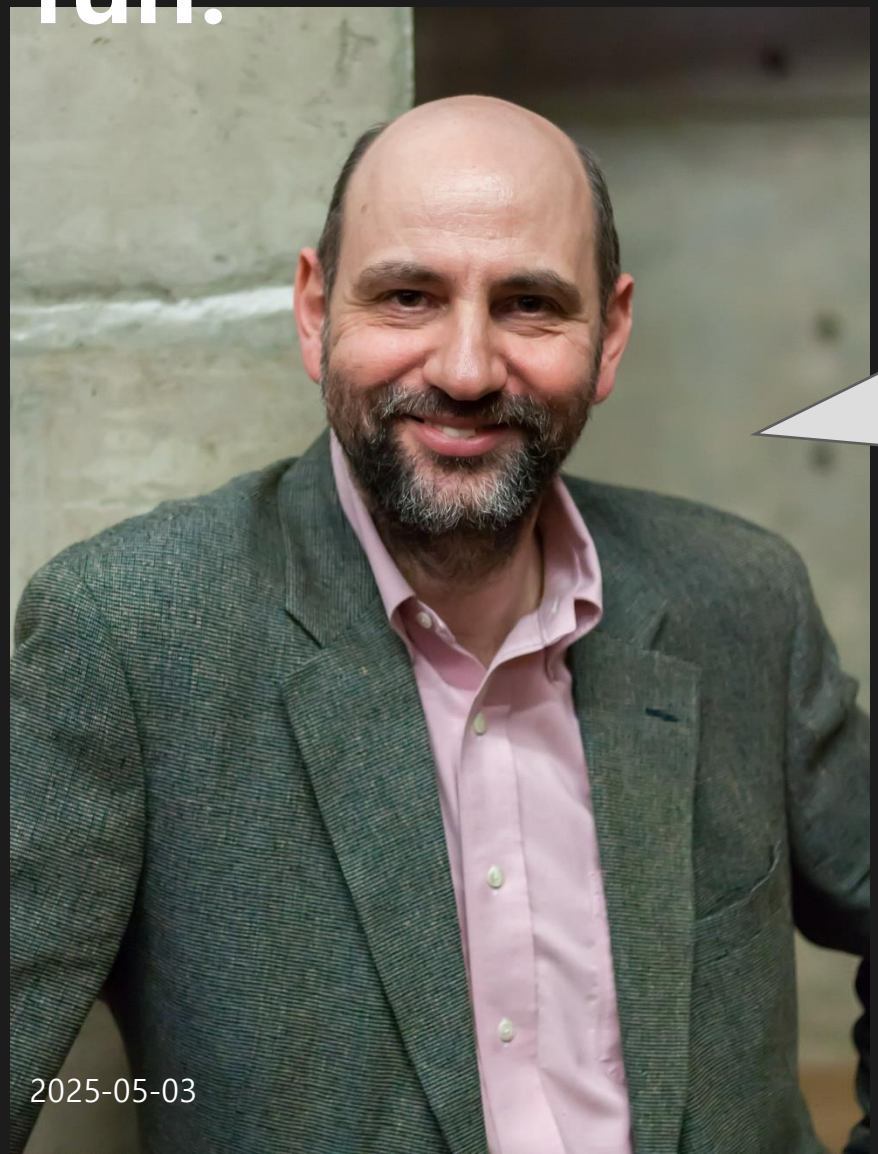
Code Change Lines: 10

Add obstacles to make the game more fun.



From now on, whether the moveRandomly() method of a car succeeds will depend on the Round.

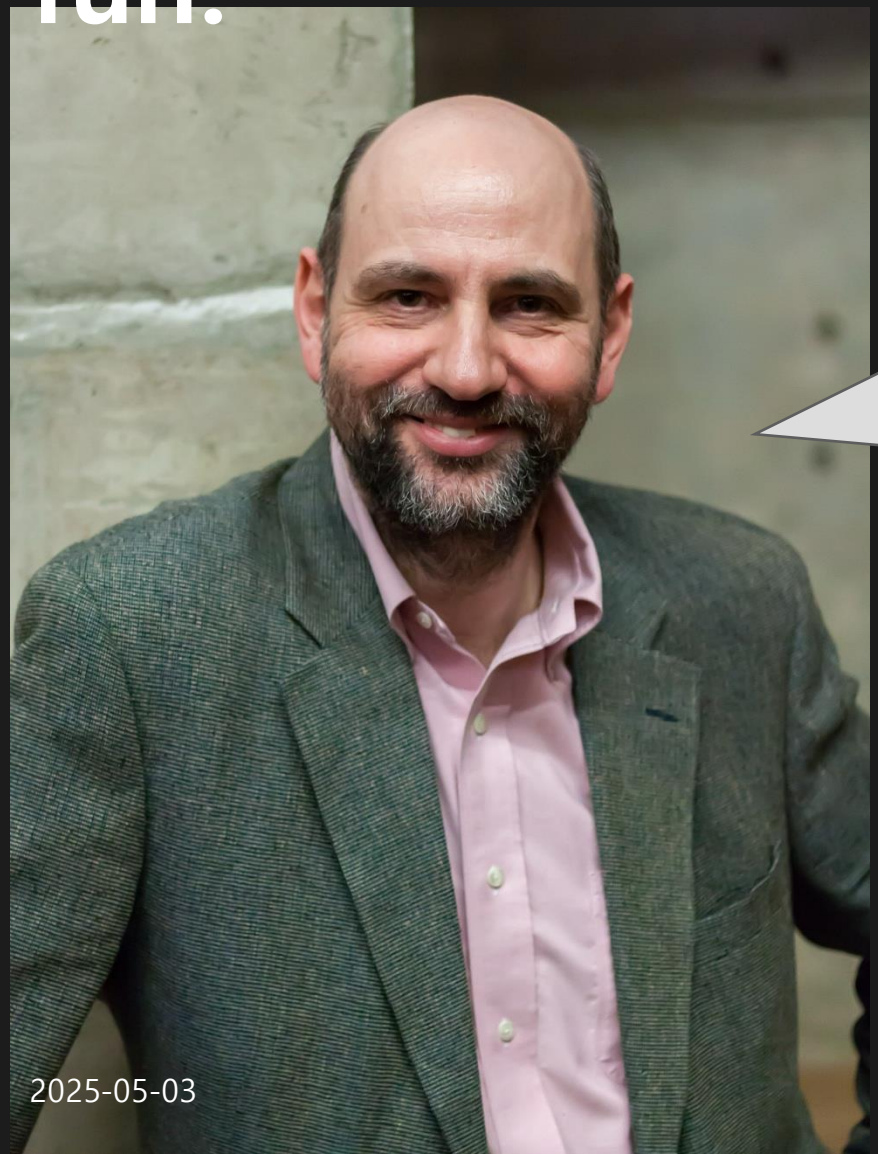
Add obstacles to make the game more fun.



Since Round is calling *Cars.moveAll(), I'll need to inject the current event (obstacle) into moveAll() as well.

- Don't process loops or filtering logic in the business layer. Use a first-class collection.

Add obstacles to make the game more fun.

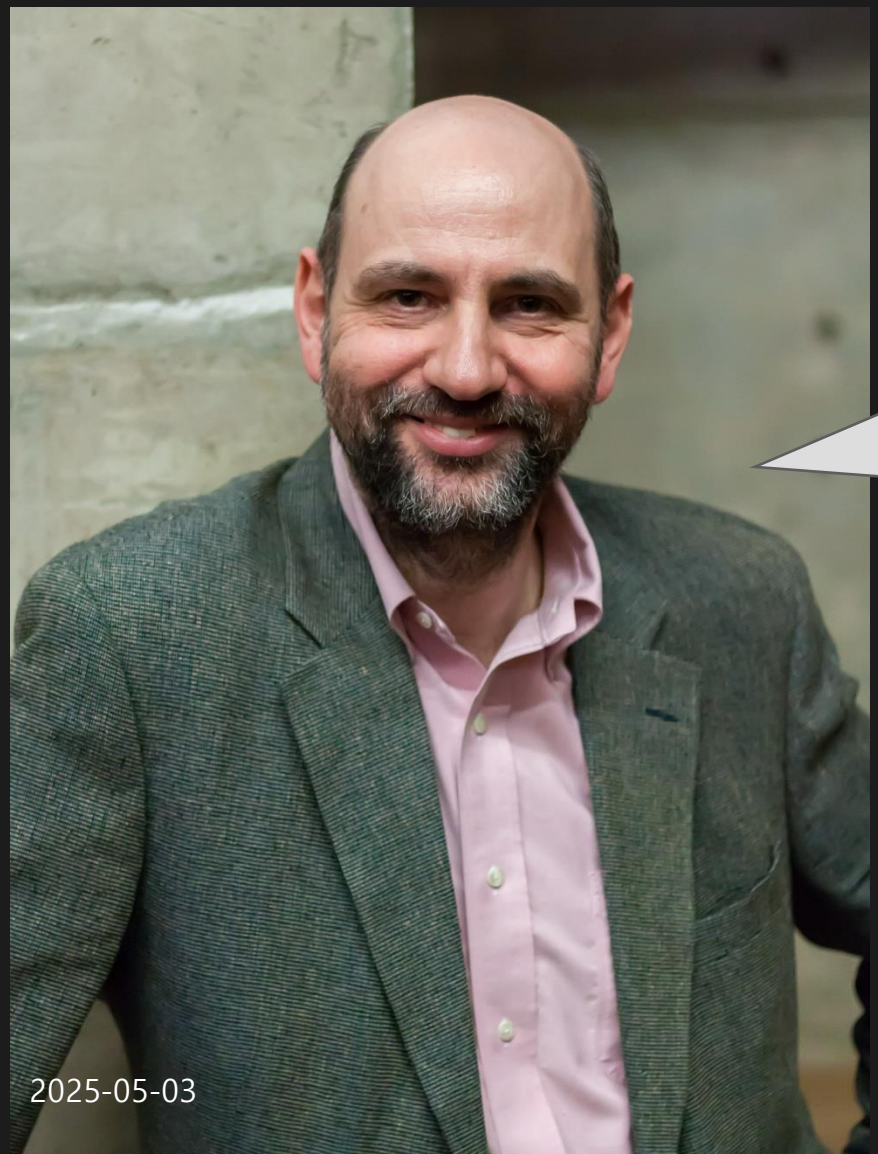


If Round decides what happens when Car meets an Event, like calling car.move() or car.accident()…
I swear I'll lose it. That's not its responsibility.

A taste of annoying customers

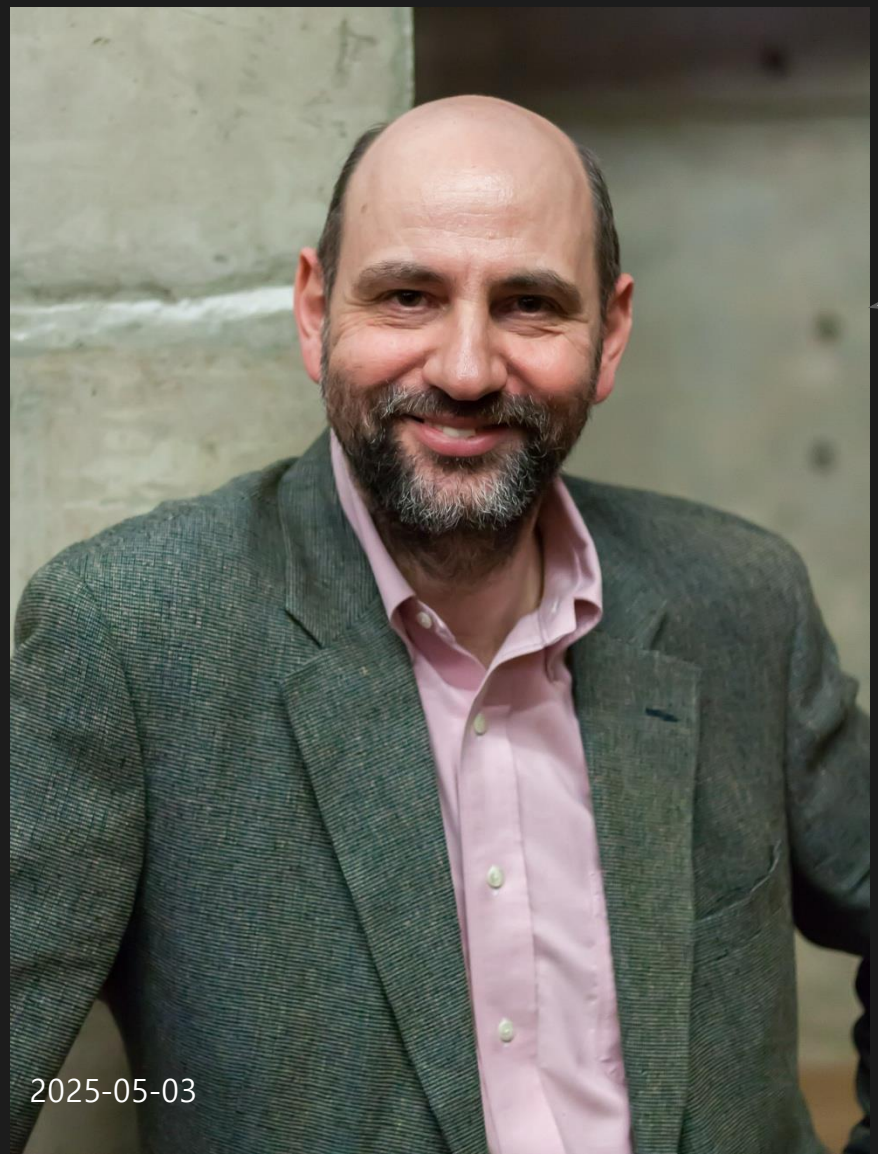
Oh, and in addition to obstacles, please add items too. Like in Mario Kart!

Please make items appear in specific rounds



Alright. I'll add an interface called RoundEvent. Both Obstacle and RandomItem will implement it.

Please make items appear in specific rounds

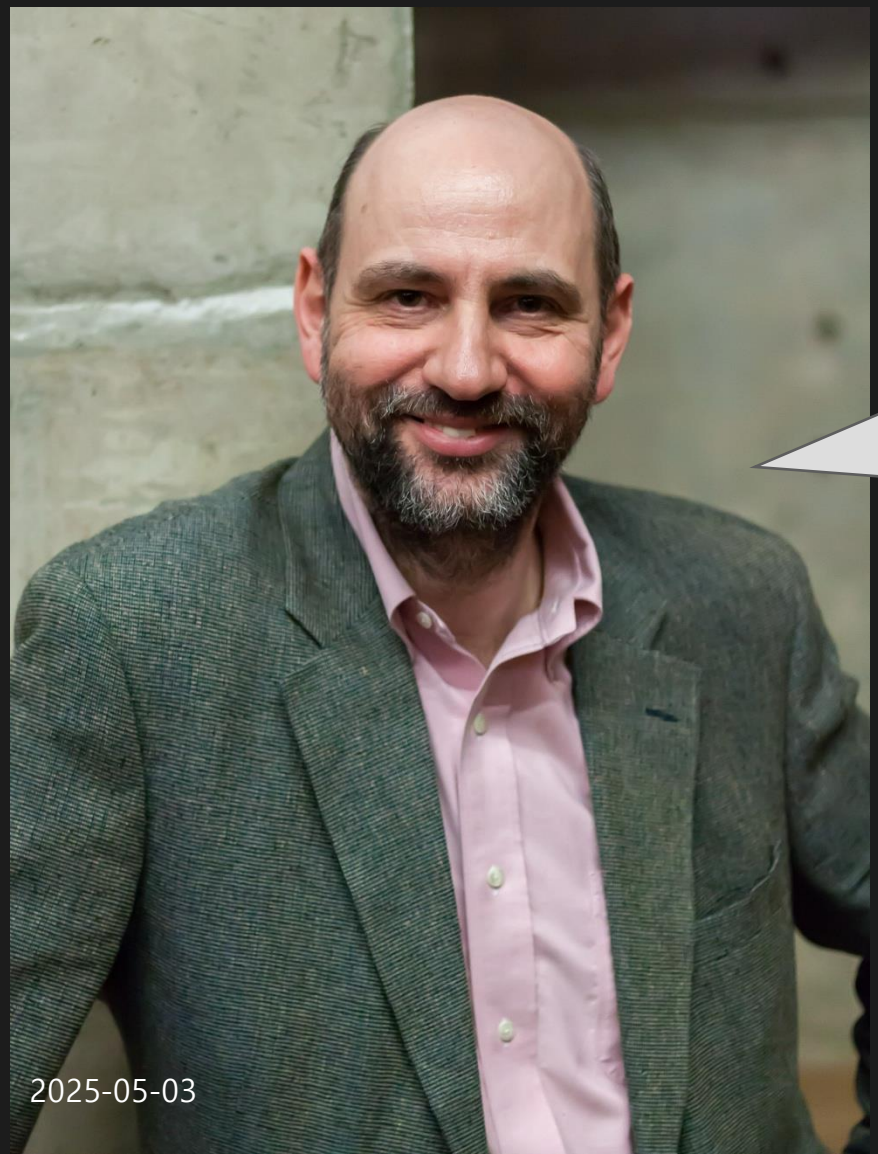


This is how I'll structure :

A taste of annoying customers

(…what are you trying to say?)

Please make items appear in specific rounds



Of course, only one event should occur per round. Got it?

A taste of annoying customers

Oh, is that so…?

A taste of annoying customers

The game's so much fun now. Let's turn it into a league mode.

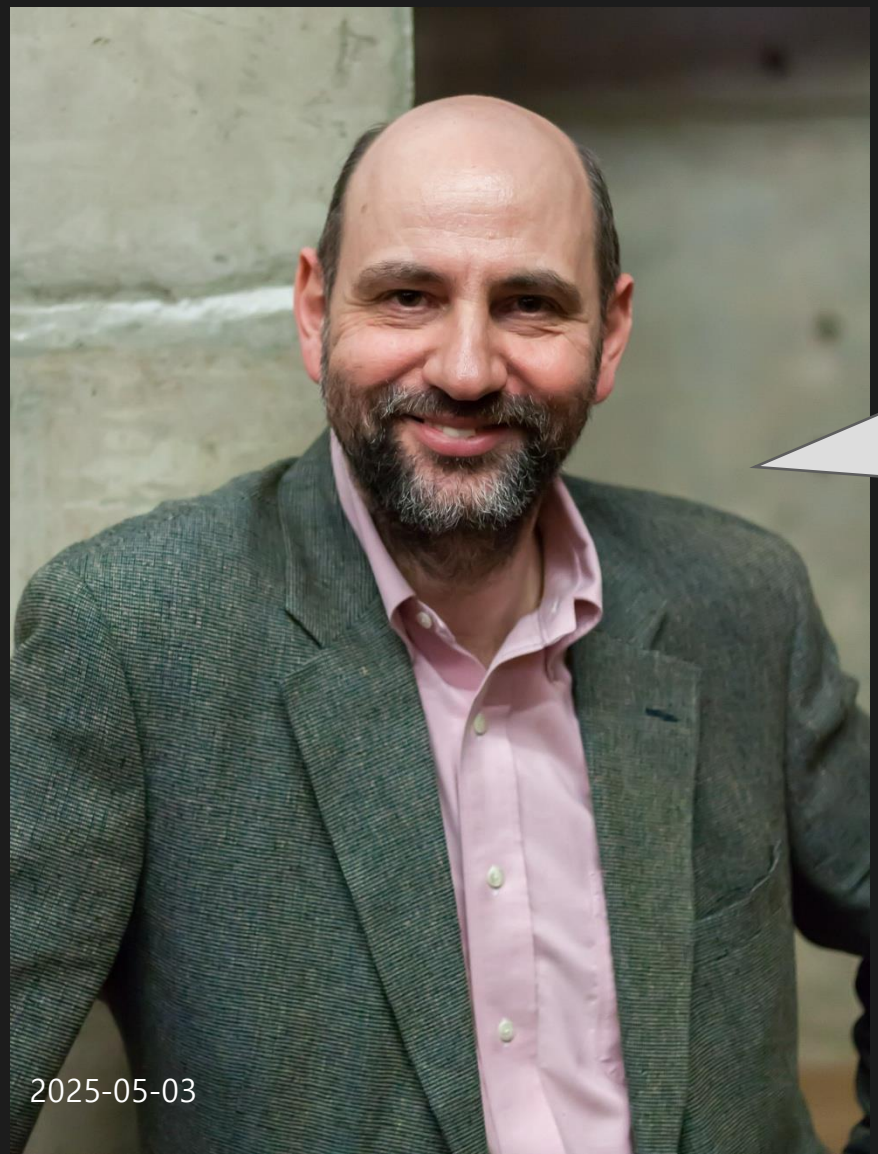
A taste of annoying customers

Hmm... the game should keep running until I say stop.

A taste of annoying customers

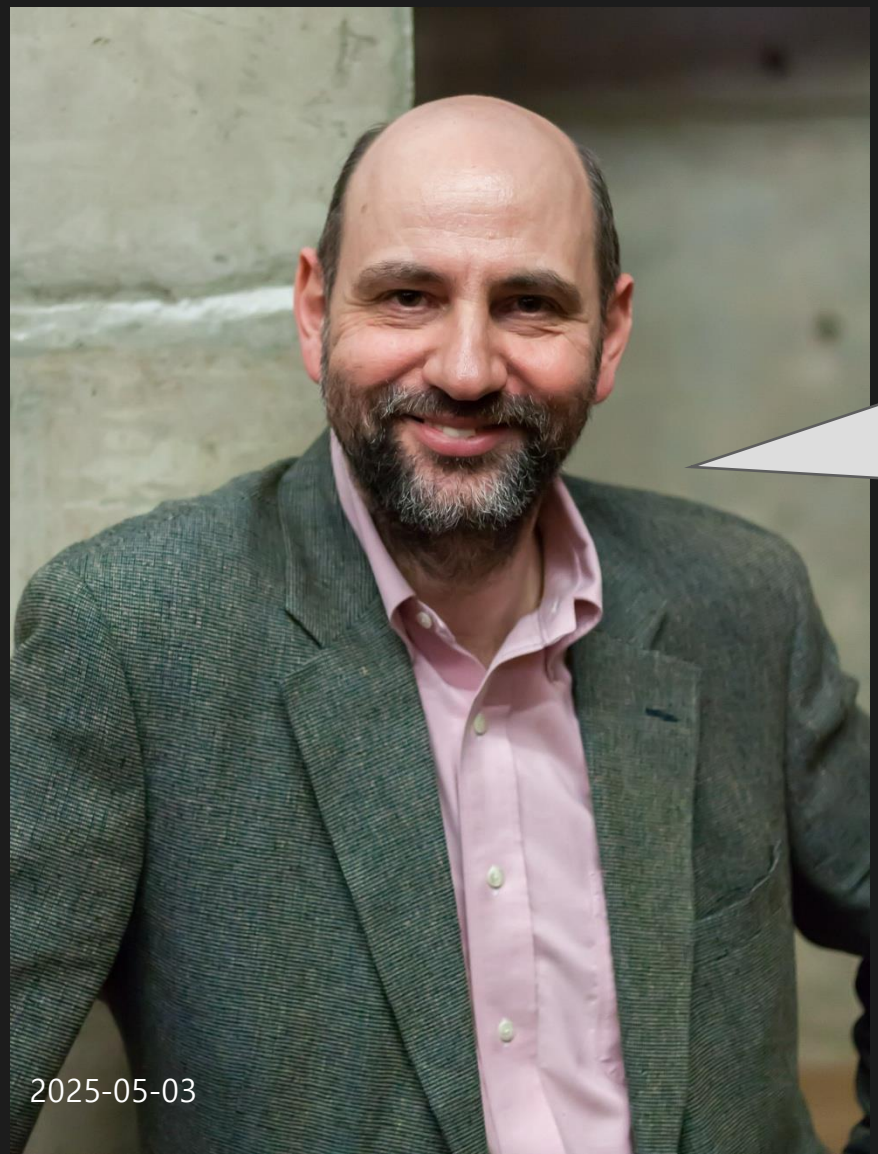
Please export the game records in Excel.

1. The game must repeat until it’s manually stopped.



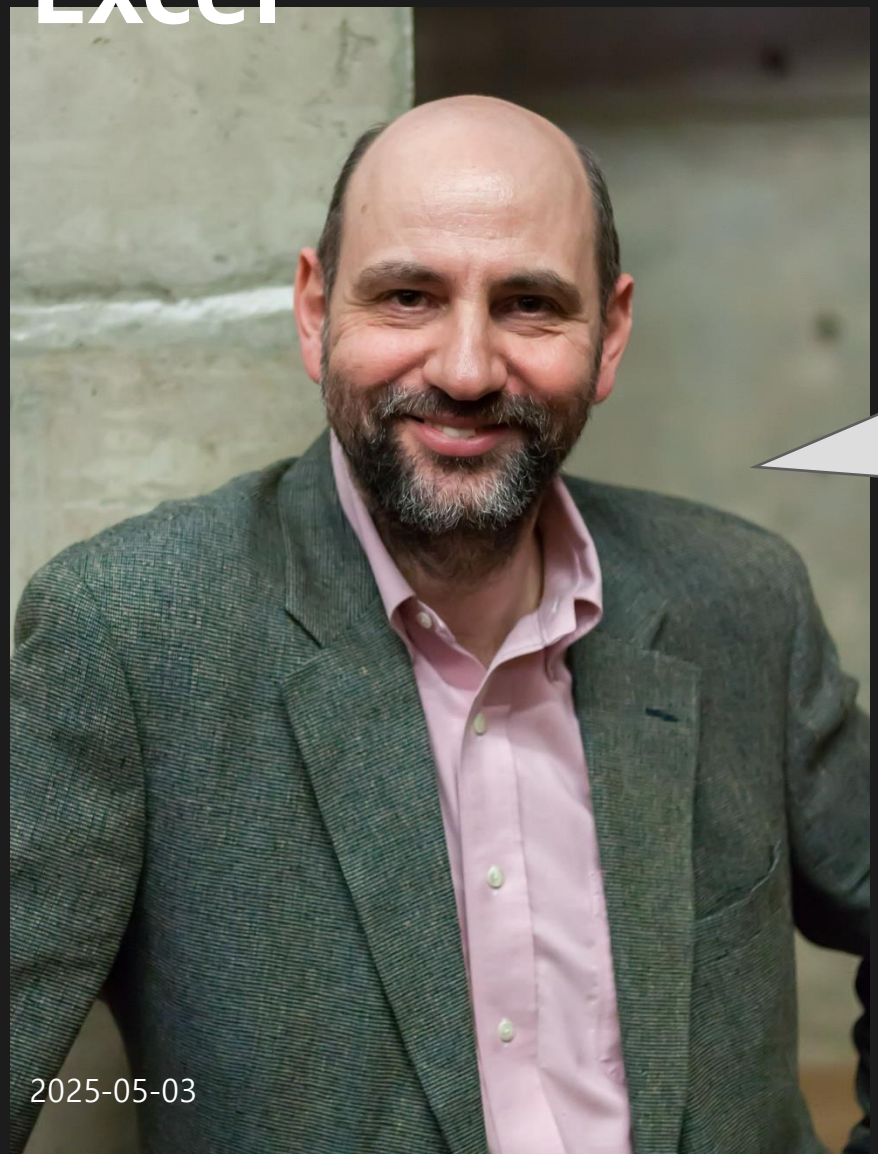
Ugh… this is giving me a headache.

1. The game must repeat until it’s manually stopped.



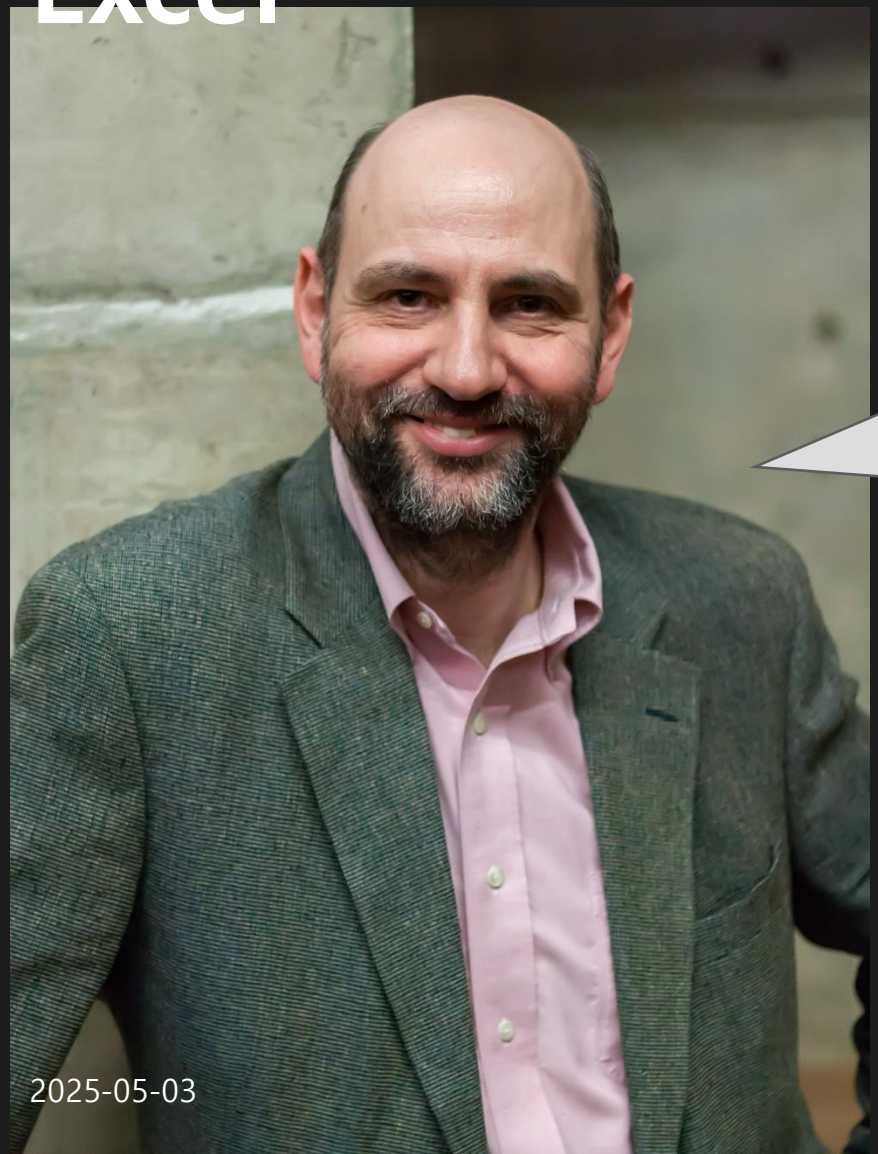
Well, I’ve already separated the game into a Racing class. Just run it in a while loop.

2. Game results must be exported to Excel



Easy. I've got a RaceResult class already.

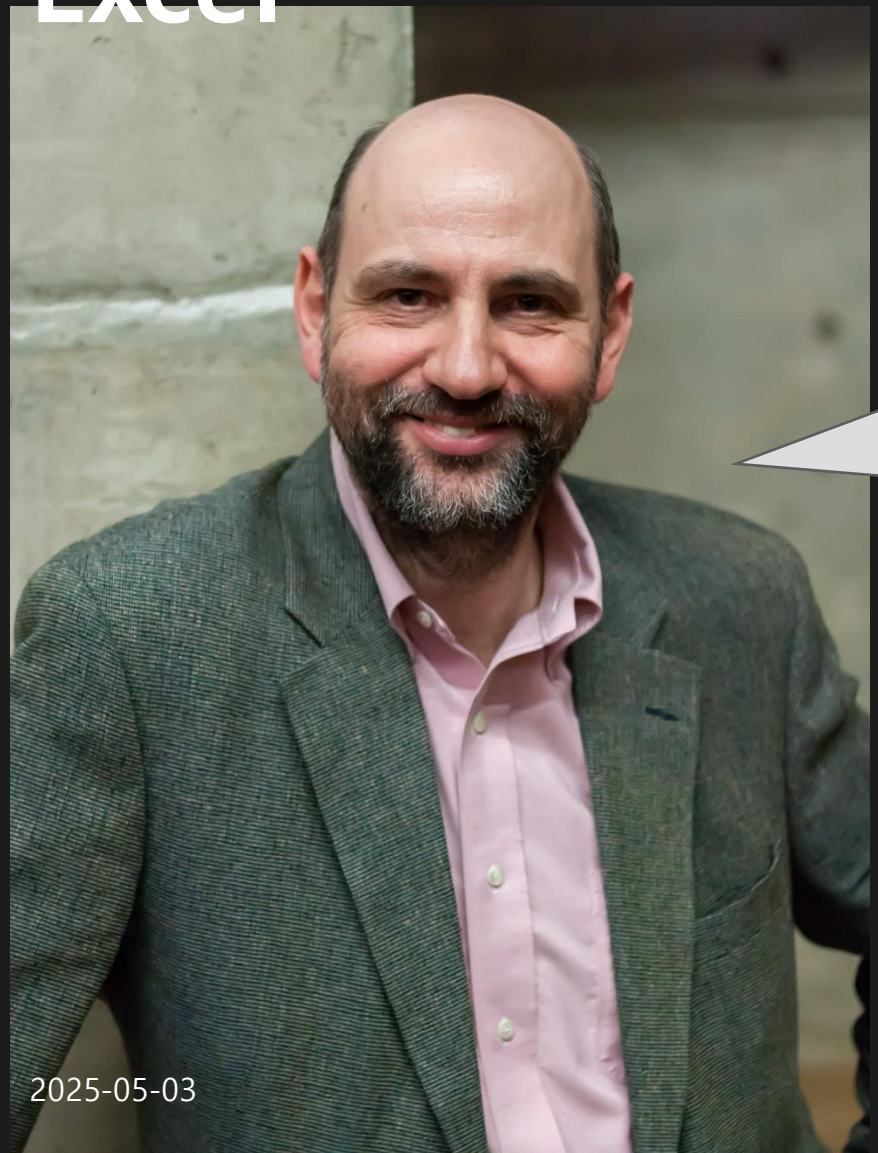
2. Game results must be exported to Excel



I'll implement an ExcelFormatter in the View Layer.

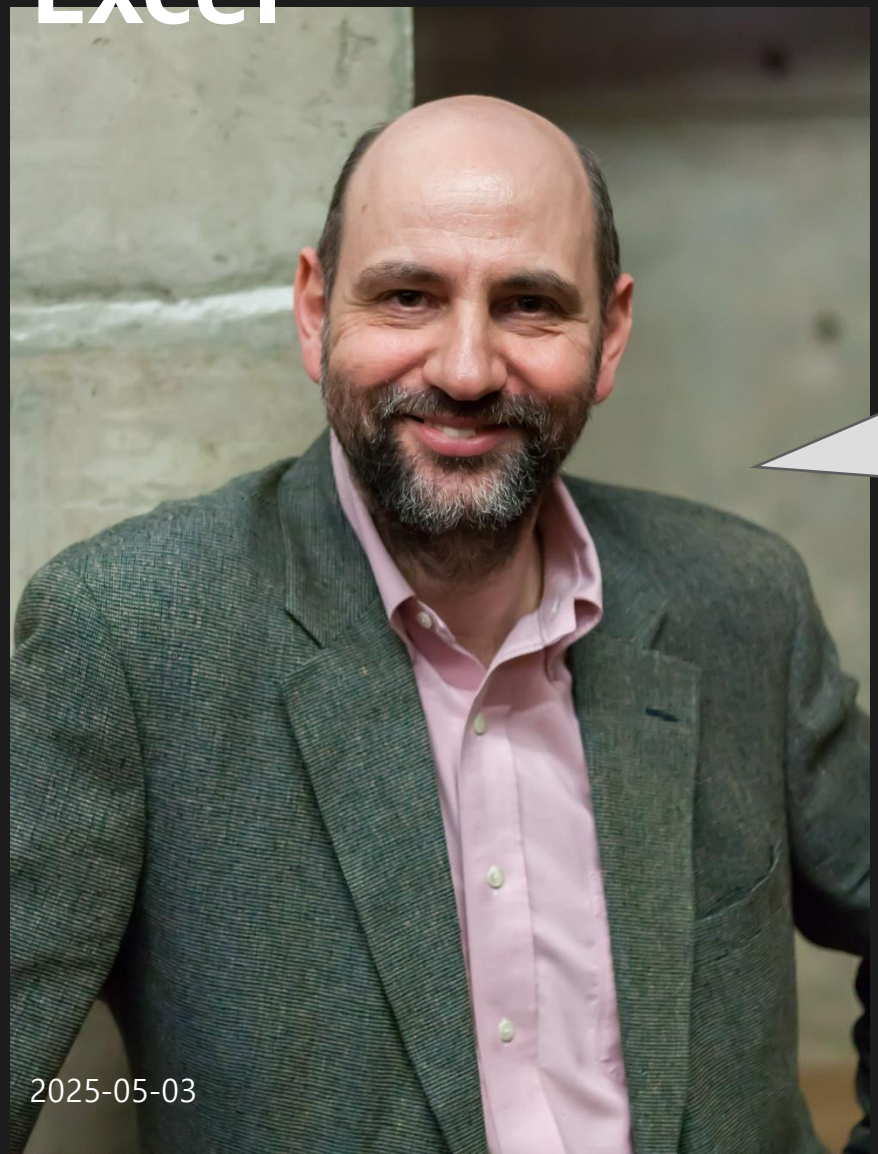
- (MVC pattern: View layer)

2. Game results must be exported to Excel



Just call the formatter inside OutputViewer.showResult().

2. Game results must be exported to Excel



And don’t you dare put parsing logic in the domain layer. It’s hell to maintain later.

A taste of annoying customers

Hmm… should we continue the rest tomorrow?

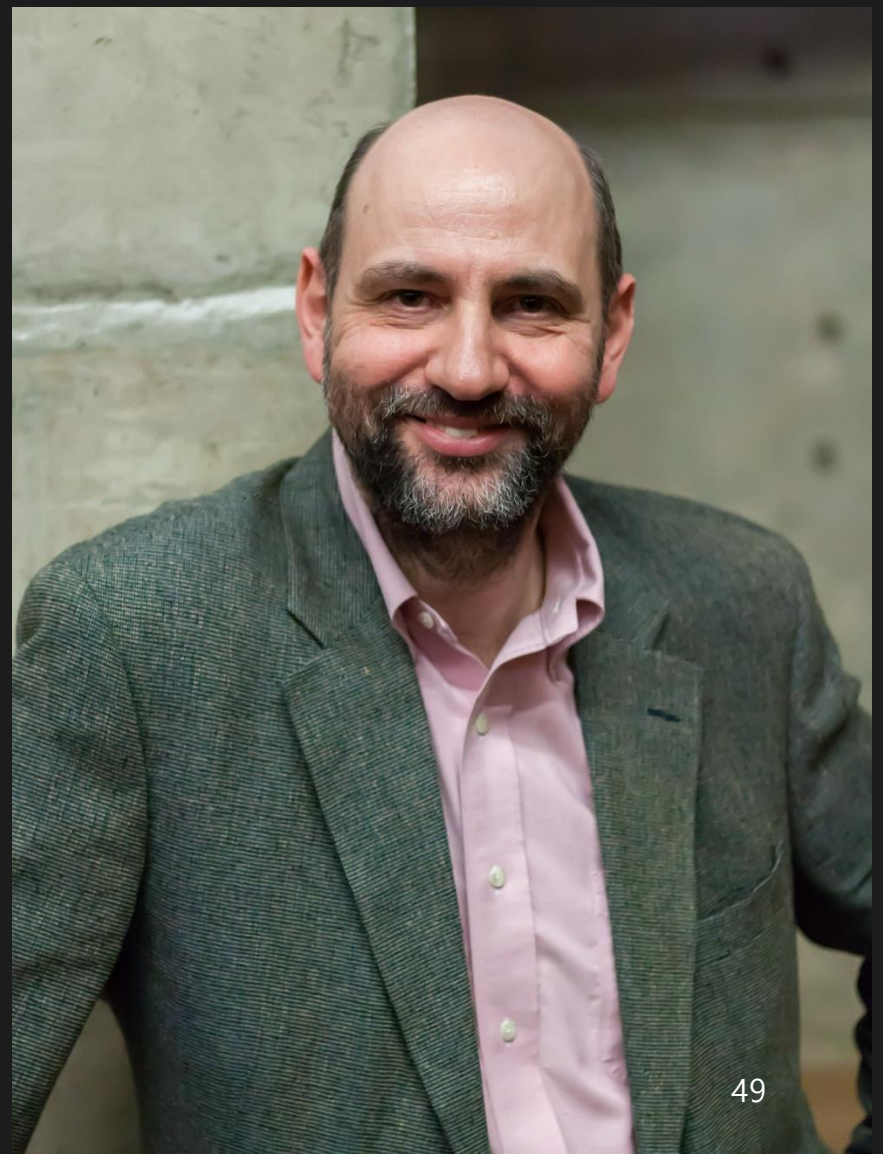
A taste of annoying customers

I really think I'm a genius planner.

End of encounter with the "lovely" customer.

What is good software design?

- “If the system is well structured, adding or changing features can be done faster and more efficiently.”
- How do you structure it?
  → “Just with experience (‘짬’).”



What is good software design?

- “If the system is well structured, adding or changing features can be done faster and more efficiently.”
- How do you structure it?
  → “Just with experience (‘짬’).”
- So here and now, to gain experience, I will conduct a live code review with you.