Twitter Thread by Nicolas Lehuen

Nicolas Lehuen
@nlehuen

I just completed "Rain Risk" - Day 12 - Advent of Code 2020 https://t.co/0wRPluJVeL #AdventOfCode

Today I learned that I really need coffee ■■ to operate properly. Made a trivial mistake and it took me forever to catch it. This would have been obv. with a statically typed lang.

Also, I'm using a notebook-style env. to play (like https://t.co/JgFUNSSRuD, here it's https://t.co/XrswSxjjwk). My take away from this fun experience + observations at work is that such notebooks are poison to the mind, fostering bad practices while not bringing much value.

I get that notebooks provide a nice environment for tutorials - you get a literate programming + a printf-debugger on steroids, which is very useful when suffering through tensor shape mismatch errors. It's useful for data science or ML 101.

But then I see people using Python notebooks to do actual work and it's horrifying to me. The natural tendency is to write notebooks as a series of cells mutating global state. So each cell has an implicit API defined by its interaction with the global state. 2/9

The API is implicitly a function of cell exec order, but then you can purposely (or mistakenly) exec cells in any order ■. And this is on top of the usual issues you get with dynamically typed languages. No one can write maintainable code this way, but notebooks get a pass. 3/9

Of course, it's possibly to organize a notebook in a proper way, to implement unit tests etc. But this pretty much kill the appeal of using notebooks in the first place. As a result I'd guess lots of workbooks that are still used for serious work are a quagmire of tech debt. 4/9

Even during a 30' coding session, you can feel the nefarious effect of notebooks. You spot a bug in a cell and fix it. Are you going to click the "Run" button just next to your fix and resume your work immediately, or click on "Runtime > Run before" and possibly wait a while? 5/9

In the midst of an idea it's hard to resist click on "Run" right there, esp. if running from the top takes long minutes. So you just mutate some global state (at least some function def) and manually follow downstream dependencies. Of course this will go wrong at some point. 6/9

For instance, you could have missed some dependency and the global state is now inconsistent, yet you keep going forward with your idea. At one point you'll realize your mistake but the harm is done, a bunch of cells are now breaking the implicit global cell + exec order API. 7/9

So not only notebooks foster bad coding practices, but they actually don't make the coding experience much more simple or reliable. The gateway drug of manually running a few cells and getting nice visualization of the output quickly devolves into an horrible mess. 8/9

So I guess the TL;DR is: don't let friends use Jupyter or Colab notebooks. They are bad for the mind and bad for the code. Also, get off my lawn■. FIN 9/9