Random talk about code!

Aras Pranckevičius, 2023 April

Who the f are you?

- A programmer. Since 1996, professionally since 2001.
- 16 years at Unity. Graphics engine and a lot of other stuff.
- Old and jaded, and I have *Opinions*™ that may or might not be correct!
- Website aras-p.info and Mastodon mastodon.gamedev.place/@aras

100% of seniors

Are just old

Or they have a handful of tricks

Effective Debugging

- Debug.Log(), Console.WriteLine() or printf() are *not* effective debugging!
  - ...except when they are :)
- Use a good debugging IDE: Rider, Visual Studio etc. Specialized tools like Frame Debugger or Renderdoc.
- Hypothesis, validation of it.
- Divide and conquer.
- Do not check multiple hypotheses at once.
- Keep notes on *everything*.
- “Surely *this* can not happen” — check it. Maybe it can?

Effective Profiling

- “I *think* this is slow because X”. Do not think, measure!
  - Results are *often* surprising even for experienced folks.
- Use good profiling tools!
- Unity has some built-in (Profiler, Memory Analyser).
- Some IDEs have decent ones (Rider, Visual Studio).
- Native code profiler can be very useful even without having engine source code!
- Superluminal is *most excellent* (check their Unity profiling).

Aside: Performance

- Computers and *ridiculously* fast these days.
- We regularly piss away 100x-1000x of their possible performance.
- Unity DOTS (and DOD/ECS in general) are an attempt at that.
  - Other approaches are possible too!
- “Practical Optimizations” talk by Jason Booth: www.youtube.com/watch?v=NAVbI1HIzCE

Use Tools in General!

- A good IDE.
- Nifty command line tools. ripgrep, fd etc.
- ChatGPT, but assume 50% of what it says is bullshit.

Intuition aka Experience

- Over time you build up “intuition”.
- “Magic” insights señors have are just because they have seen a similar situation 10 times.
- Not sure if there are shortcuts? It might take 10+ years.

Important vs not

- A lot of programmer flamewars are a waste of time.
  - Tabs/Spaces, Vim/Emacs, OOP/DoD, NULL/nullptr, */&, coding style, D3D/Vulkan, ...
- Too much worry about “future extensibility”.
- Too much worry about “clean code”.
  - Users (players) don’t care about your code.

Larger Picture

- Keep a larger picture in mind.
- This change/feature/fix/system/program - what is the *actual* goal of it?
- How does it interact with things around it?
- What are the failure cases?
- What else could be affected by this?
- “Leave place in better shape than you found it”

Effective Source Control / Comments

- Explain *why*, not *what*.
- In a year you will have forgotten “why”.
- Write down why something “obvious” is not done.
- Learn your VCS! Branches, blame/annotate, bisect, etc.

Root Causes

- Get down to the root cause of things (bugs, behaviour, …)
  - Within reason :)
- “Dividing by 2 makes the issue go away” — ok, but *why*?

Curiosity

- “Hmm this feels slow” or “Huh this looks strange”.
- Investigate!
- And then do something about it.
  - Or get someone else to do something about it.

Managing Time

- Get out of “I’m stuck” without wasting too much time.
- Context switching.
- Interleave high-focus and easy/trivial tasks.
- Take a break!

Ask me 5 or more questions!