A Physical Metaphor for Quick Fixes and Root Cause Analysis

By Steve Berczuk - July 14, 2020

If you deal with legacy code you've likely found yourself struggling to debug and fix a mysterious, intermittent problem. Along the way you may have discovered some code that didn't quite make sense. While it's tempting to express dismay over the decisions a previous developer made or to assert that you'd never fix a problem in that manner, you probably realized that they didn't expect their hack to endure, and you might have done something similar. While going through some old books, I came across a story that seems like a great metaphor for this kind of legacy code experience. In David Owen's book about renovating an old house, The Walls Around Us, he shares a story about fixing a leak coming from the ceiling. Water issues in houses can sometimes be very much like debugging intermittent software problems: you only see the symptoms occasionally, and the visible indications don't always point to the source of the problem. When water started leaking from the ceiling in Owen's house during one particularly rainy season, he discovered that the source of the leak was an old overflowing bathtub in the attic. A previous homeowner, having given up on finding the source of a leak, hauled an old bathtub to the attic to catch the drip. This worked well for many years, until that particular rainy season when the rain overcame the rate of evaporation.

One can imagine that the previous homeowner didn't really believe that a bathtub in the attic was the right fix. But given the choice of seeing the leak continue and deferring other needed work on the house while looking for the source and a fix that would work, a quick fix didn't seem all that bad. Since the "fix" worked for many years, it's easy to see why finding the root cause never became a high priority. This is reminiscent of my experiences working with legacy code. Either we discovered code that managed to hide a problem for a long time, or worse, we ended up doing that sort of fix—a well-intentioned solution because there was no obvious way to fix a problem without a complete teardown. In the moment the team decided that a proposed quick fix was perfectly reasonable—ingenious even! But while we thought our decisions balanced good, practical, and expedient in the moment, in retrospect they may have been the code equivalent of the bathtub in the attic. Something that hides the problem will catch up with you. These kinds of changes aren't always the result of thoughtlessness. Cost and benefit are always things we consider, and a quick fix may be the thing to do in the short term. The problem is when you don't follow up. Any expedient fix should be treated like technical debt and made visible—with a plan to evaluate and perhaps address it later. Items in your issue tracking system, code comments, and tests are all ways that you can make it easier to find the right fix for the problem you worked past. While not desirable, sometimes a bathtub in the attic happens. Try not to leave it as a surprise for a future occupant of your code.