A Poor Workman Blames His Tools

Measure Twice, Cut Once – September 2022
Rob Carver

This familiar saying suggests that one who lacks sufficient skill to execute a task well tries to shift the blame to the tool. A web search reveals that it is a very old expression, possibly evolving from a 13th-century French proverb *mauveés ovriers ne trovera ja bon hostill*, 'bad workmen will never find a good tool'. The 17th century British poet George Herbert wrote "An ill labourer quarrels with his tools." Perhaps a more modern rendering would be "Novice woodworkers blame their tools."

When something goes wrong, it is tempting to find fault with a tool rather than the worker. Skilled, knowledgeable makers understand that the mistake was most likely of their own doing. One might argue that an important milestone in one's development as a woodworker is the ability to look inward and acknowledge one's own errors. At one time or another, we've all made an error and search for somewhere to lay the blame when the obvious culprit is at the other end of the tool. No accusations here, but have you ever slammed a tool down on the bench or flung one across the shop in frustration?

This tendency could underlie the prevalence of tool ads and blog posts about tools that promise to improve your projects, much like that next golf club that will eliminate your wicked slice or the diet program that will shed pounds effortlessly. To be sure, using the right tool for the task is one key to success, but usually a woodworking failure is not the fault of the tool, but of the woodworker.

Early in my career I wrote computer programs for a living. In the early-to-mid-70's, one popular term that made the rounds was the ideal of the "egoless programmer." An essential task in programming or coding is debugging – searching through a chunk of code that does not work as intended. The simple goal is to find and correct the error. Debugging your own code is painstaking work, and often bumps up against the issue that the creator of the code was the creator of the bug. The technical, logical challenges of debugging can be significant, but sometimes miniscule in comparison to the psychological barriers. Ego impedes the work, hence an ideal programmer was egoless.

Later, when I transitioned to teaching, I taught coding. I clearly recall a visit from a frustrated student who came to me with a stack of green-bar paper and blurted out "YOUR computer won't run MY program correctly!" Later still, while teaching statistical analysis with open-source software that had weekly user-contributed updates, I cautioned my classes that sometimes updates could cause code to fail, even if the very same code had run previously. That little speech opened the floodgates, and students started to write off every one of their own mistakes to those darned updates.

A coder might want to attribute a problem to changes made by the incompetent stewards of the language, or more commonly to "User Error" – there's nothing wrong with my code, but

that dumb user did something foolish. Doing research for this essay, I ran across the PEBCAK error, which is new to me: “Problem Exists Between Chair And Keyboard”.

Whether the tool in question is a laptop, a chisel, or the latest CNC machine, you don’t have to be a psychologist or philosopher to know that it is hard for us to accept responsibility for our goofs, or that our instinctive response to an error is to scan the environment. Flawed tool, network service interruption, inferior steel, PEBCAK – we have an endless collection of usual suspects when things go awry. And yet, part of woodworking maturity is to look in the mirror to locate the root cause of a problem.

To misquote the Bard, “The fault, dear Brutus, is not in our tools, But in ourselves.”