Microspeak: Spinning up or kicking off a build

devblogs.microsoft.com/oldnewthing/20131112-00

November 12, 2013

Raymond Chen

Remember, Microspeak is not merely for jargon exclusive to Microsoft. It's jargon you can employ to sound more like an insider.

Most of the time, a fresh build of Windows is produced every day. When the product is nearing a release, train service gradually declines, with faraway branches losing service first. Eventually, the train stops running altogether, and the release branch remains stable for extended periods, not accepting any *payload*. This allows the test team to run their supersized test suites, the ones that take days to complete. (Known in the jargon as a *Full Test Pass*, or *FTP*. Sadly, this is easily confused with the File Transfer Protocol, also called FTP.)

Of course, the Full Test Pass may discover some problems, and if one of them is considered serious enough, a fix is applied to the release branch, and then a new build *spins up*.

The metaphor here is that the computers that produce the build have been sitting idle for so long that the imaginary turbines have stopped spinning. To build another release, the turbines need to get up to speed.

The Microspeak phrase *spin up a build* applies more generally to the concept of *producing a one-time build from a branch that has not had a build in a long time*. The branch may not have had a build because it has stopped accepting changes, like our release branch in the example above. Or it may not have had a build because it has intentionally been abandoned.

An example of the latter case is a special branch for experimental work that will never go into any official release. The build lab may be under instructions not to produce a build from this branch because the developers working in the branch can just rebuild the components they are experimenting with. Or because the branch is in a constant state of flux, so trying to produce a build will usually result in chaos. Or because the opportunity cost of producing a build out of this branch is denying a build to another branch due to limited hardware resources, and the people in the experimental branch are willing to take on some of the work themselves in order to free up resources for the other team.

If for some reason, the experimental branch would like the build lab to produce a build, they can ask the build lab to *spin up a build* for the experimental branch. (Presumably with a phrase like "when you get the chance" so that the build team can *kick off* the build at a time when demands for build machinery are low, like in the early afternoon.)

Kicking off a build is the jargon term for starting a build, presumably inspired by the *kick-off* which starts many sporting events.

Most regularly-scheduled builds occur overnight, and you might ask, "When does our build kick off?" This means "What time does the build lab start building our branch?" You would ask this if, for example, you want to know how much time you have left to get your fix signed off if you want the fix to go into tonight's build. If the build kicks off at 5pm, you might say, "There's no way I can get it done by then." But if it kicks off at 7pm, you might say, "Maybe I can convince my testers to drop what they're doing and test this fix, and then I can address the problems they find and get the fix signed off by 5pm. That leaves two hours for the change to make it through the gated check-in queue." (When a check-in successfully exits the gated check-in queue, it is said to have *cleared the queue*.)

Even if the build normally kicks off at 5pm, you can ask your branch manager, "Can you delay kicking off the build until 7pm? I have an important fix that we really would like to get into tonight's build, and there's no way it'll be ready by 5pm, but I think I can make it by 7pm." Of course, every hour you delay kicking off the build is an hour later the build becomes available in the morning, so the branch manager needs to balance the cost of delaying the release of the build with the benefit of picking up this fix.

That's a lot of build-related jargon for one day, but these are terms you need to know (because you will hear others use them), and if you dare, you can employ them yourself to sound more hip and with-it.

Raymond Chen

Follow