Stop merging if you need to cherry-pick

devblogs.microsoft.com/oldnewthing/20180709-00

July 9, 2018

Raymond Chen

The VSTS team wrote a response to my series from a few months ago titled Stop cherry-picking, start merging.

> **Stop merging if you need to cherry-pick**
>
> As the stewards of Visual Studio Team Services's Git server, we read Raymond's stop cherry-picking, start merging series with great interest. When Raymond pays attention to your area, you should probably pay attention to what he has to say. After finishing the series, we both agree and disagree with his conclusions.
>
> Given the constraints Raymond's team works under, we think he's found a pretty good solution to some very real problems. Windows, for historical reasons, has a lot of long-lived branches that need to merge into each other pretty often. If you need to fast-track a fix from one branch into another ahead of the official integration schedule, you're definitely setting yourself up for the kinds of conflicts Raymond writes about.
>
> **But**… if you aren't Windows, you probably don't have this problem. Over in VSTS, we use and recommend a trunk-based development model with few long-lived branches. While our "Release Flow" model does include servicing branches for some releases, those branches will never merge back together. Thus, we don't encounter the merge conflicts and silent work reversions that Raymond's team does.
>
> In a way, the solution is almost as painful as the problem. You have to know ahead of time what branches you're going to cherry-pick your commits into. If you don't, you can make quite a mess of your Git graph. And if anyone on your team doesn't fully understand the history contortions this workflow involves, they can mess it all up for you. For those reasons, and because we expect it's rare outside of Windows's workflow, we don't plan to put any features into VSTS to automate this.
>
> One other note: think twice (or maybe three times) before you `git merge -s ours` under any circumstance. While it's the right thing here, you're *intentionally throwing away someone else's work*. We've fielded innumerable customer tickets of the form, "Git lost my work". In the vast majority of cases, the culprit was someone resolving merge conflicts by throwing away work. Git didn't lose your work — you asked it to forget your work!
>
> Thanks to Raymond for writing this series, and for allowing us to pile on. The Windows team has been a great partner in helping us make our server (and indeed all of Git) scale to insane sizes and workflows. We're always considering new approaches to Git which might be applicable to the wider community.

I extend my thanks to the VSTS team for providing their perspective.

As the VSTS team notes, the problem case is where you cherry-pick between two branches that will eventually merge. If the two branches never merge, then there's no need to get all fancy with your cherry-picking.

Raymond Chen

Follow