Snapshot

Who can use this?
Core
Plus ✓

Snapshot provides a summary of an individual developer's work patterns over a period of time. Snapshot gives developers feedback they can use in one-on-ones or reviews.

Snapshot works best when tracking how an engineer's position changes over time, rather than their position at any specific moment. We recommend two week time-spans so you have a clearer picture of the different ways a developer contributes to the team. A developer who makes contributions one week might contribute in other ways the following week.

In this article
What do the different quadrants mean in the Snapshot report?
How does the feedback work in the Snapshot report?

What do the different quadrants mean in the Snapshot report?

The chart plots an engineer's contributions across 2 axes:

- **Impact** is the severity of edits to the codebase, as compared to repository history.
- **Rework** is code which was thrown away shortly after being written.
Tip: The bottom two quadrants usually are typical for part-time engineers, team leads, managers, QAs, etc. If you have full-time developer fall in the bottom two quadrants—especially the bottom-left ‘could be stuck’ quadrant—they may need some help.

Exploring (a.k.a. discovery)
Developers in this quadrant might be working with a new language, experimenting with a new type of feature, or trying out many implementation paths. Occasionally, engineers in this quadrant haven't been provided with clear specs or requirements, which is why they have high rework.

Prolific
Developers in this quadrant are making exceptional contributions. If the developer has the correct requirements and are building the right thing, interrupting them might be costly. They are building impactful code that doesn't rework.

Could be stuck
If an engineer is in this quadrant, they might need some help on a feature. It might be worth checking in, especially if you see them in this quadrant for weeks. You don't want full time engineers to be here for very long, so you can use this report to check in and help before it's too late. Sometimes QA and Product Managers appear in this quadrant. If you'd like to exclude these irregular users, see Excluding users from reports.

Perfectionism
This is a quadrant you might expect managers, team leads, and other senior roles who are familiar with the codebase. They tend to have lower throughput because they are helping others or leading a team. A full time engineer wouldn't typically remain in this quadrant since they're not making a huge impact to the codebase, but most of the work they check in doesn't get reworked.

back to top
How does the feedback work in the Snapshot report?

Snapshot gives 10 possible pieces of feedback. The ranking functions sort into a waterfall with these possible outcomes:

1. Check in your latest progress every day.
2. Spend a little more time coding each day.
3. Take on some more important tasks.
4. Break large work items into pieces.
5. Develop a clear implementation path before starting work.
6. Balance large refactors with work on new features.
7. Take on some more difficult work.
8. Help the team pay down technical debt.
9. Review code only after making progress on your own items.
10. Honestly, we couldn't find any suggestions. Keep kicking butt.

Here are some examples of how that feedback might be presented:

- An engineer might rework at an unusually high rate relative to the team, so **Develop a clear implementation path before starting work** may be relevant. However, if they are also checking in abnormally large commits, they would receive the higher-priority **Break large work items into pieces**.

- If an engineer's work signature contains mostly low-impact commits, they may receive **Take on some more difficult work**, encouraging them to dig in and take ownership of larger features.

- If an engineer falls substantially lower than the team average for meaningful active days, they may receive **Take on some more important tasks**.

Why do I see a lot of **Check in your latest progress every day**? Some of what is at play is Flow's bias that progress should be committed and pushed on a daily basis. We've found that regularly committing and pushing code in small bites (opens in new tab) is good for productivity across the 20+ million commits we've analyzed.

As a result, anyone committing code less than three days per week receives this feedback in the waterfall logic, regardless of the team average.

If you need help, please email Pluralsight Support (opens email form) for 24/7 assistance.