Scrum Events

**[Problem Statement](#_yggcimfz9uew) 1**

**[Definitions](#_xskirq9pe08w) 1**

**[How to run your own experiments](#_5lg66kso3mk) 2**

**[Events](#_fg1bh510sgx7) 2**

**[Product Backlog Refinement](#_8y0lwo6h5jn) 2**

**[Sprint Planning](#_plnbg1kyntwz) 3**

**[Daily Scrum](#_y3c34w9d8c0t) 5**

**[Sprint Retrospective](#_83c3ioxu6y5x) 6**

**[Sprint Review](#_t4zowofhtlls) 7**

# Problem Statement

This document describes the various Scrum events and for each of them, it defines the problem statement, the objective of the event and other information useful to achieve the objective.

We wanted to write this document mainly for three reasons:

* We want to share best practices based on our experience
* We want all teams to have a shared starting point from which to experiment
* We want a written document to allow everyone to challenge our ideas and, in so doing, to share learnings with the entire product and tech team

# Definitions

|  |  |
| --- | --- |
| **Product Backlog** | A living documentation, single-ordered list of what is needed to improve the product in the future and a single source of work undertaken by the team. |
| **Sprint Backlog** | Usually it is formed as a kanban board that captures a real-time picture of a team's progress. It contains the sprint goal, an agreed selection from the product backlog. The Sprint Goal is an objective set for the Sprint that can be met through the implementation of Product Backlog. |
|  |  |
|  |  |

# How to run your own experiments

This document is not supposed to block you from deviating and finding ways that make your team more effective. This is supposed to be a “baseline” from which to start. Feel free to find your own path, but by being mindful of deviations, we hope it will be easier to come back to this baseline if things don’t work out.

# Events

For each event:

1. What’s the problem statement
2. Who should attend
3. DOs and DONTs (e.g. no need to score during grooming)
4. FAQ / Lessons Learn

These are the events we follow in our Product Development.

## Product Backlog Refinement

*Used to be called Grooming*

|  |  |
| --- | --- |
| **Problem Statement:** | PM writes user stories for engineers. Stories are a tool of communication between PM and engineers so it’s important that engineers have the opportunity to review stories before the sprint starts.  Without shared understanding, we risk implementing the wrong thing, wasting effort, and having to rework the implementation to get it right.  In this event, selected devs and QAs read the story and provide feedback / clarification questions to the PM so that in the time between Refinement and Planning, the stories are fully formed and ready to be accepted by devs without any further clarification. |
| **DOs and DONTs:** | **DOs:**   * Assign complexity points to stories which are ready; * Discuss edge cases and challenge the PM if there seems to be holes in the overall logic.   **DONTs:**   * Assign complexity points to stories in which the PM has not flush out all details. |
| **Attendees:** | Tech Lead, Product Manager, Designer (for UI teams), Scrum Master, a few Developers and a few QAs (we want to limit the impact on productivity) and also Subject Matter Experts or Stakeholders if required. Note that stakeholders’ presence should be an exception. If this is common, the PM should spend more time to better understand the problem statement. |
| **Activities:** | **Before:**   * Product Manager defines what product backlog/ideas need to be discussed; * Tech Lead invites few Developers to represent all the skill sets in the team; * Product Manager invites Stakeholders, if required; * The Product Manager / Developer invites Subject Matter Experts, if required.   **During:**   * Attendees collaborate to add more information to the product backlog being discussed; * Attendees collaborate to break down the Product backlog into smaller chunks of deliverables.   **After:**   * PM works on the gap(s) which have been highlighted during the meeting; * Act on any other action item (e.g. developers need to find more information on an upcoming activity). |
| **FAQ/Lesson Learned:** | * The Product Backlog Refinement activities should focus on and refine initiatives related to the upcoming sprint (not activities in the Sprint Backlog or in the running Sprint); * The PM should order the Product Backlog before the event to help the discussion (Note that feedback from TL and engineers might prompt the PM to change order); * Refinement is an activity (not necessarily an event): the team can do it asynchronously before the event if possible; * **Be focused - it should be a rare exception for a story to pass grooming and then to be rejected at planning.** |

## Sprint Planning

|  |  |
| --- | --- |
| **Problem Statement:** | Every single sprint must be valuable. This means that the work completed at the end of each sprint should be an increment that can be released to production. During Sprint Planning the whole Team collaborates to define a Sprint Goal. Everyone is welcome to comment and suggest, but the responsibility for defining a Sprint Goal is with the PM.  The Sprint Goal is the objective the team sets for the Sprint. If, during the sprint, the team realises that the Sprint Goal is at risk, they should drop other non-critical development activities and help the team achieve the Sprint Goal. For example, stories can be dropped, but not critical bug fixes.  During Sprint Planning the PM presents the stories and answers clarification questions. Devs assign complexity points to stories that could be not evaluated during Product Backlog Refinement due to missing information.  Initiatives that are considered within the capacity of the team for the Sprint are selected.  Naturally estimates cannot be always accurate, but the TL is responsible to track the accuracy of the team estimates and if the team tends to be always overestimating or underestimating, the TL should implement corrective actions. |
| DOs and DONTs**:** | **DOs:**   * Take time to read stories in detail; * Ask clarification questions; * Understand how the user flow (or the API response) will change when the story is completed; * Define a clear sprint goal; * Commit to an amount of work which the team is confident can be delivered; * Plan how the team will deliver the Sprint Goal (the PM is not needed during this implementation phase of the meeting); * Agree on a clear definition of done.   **DONTs:**   * Get stuck in a long conversation about the exact complexity of the story. |
| **Attendees:** | Dev team, QAs, PM, Designer (for UI teams), TL, SM, and stakeholders (if necessary). |
| **Activities:** | **Before:**   * Product Manager invites Stakeholders, if required; * The Product Manager / Developer invites Subject Matter Experts, if required; * The Product Manager acts on all the clarifications raised during Product Backlog Refinement; * The designs are ready (for UI-related work);   **During:**   * Developers estimate stories which are not yet estimated; * Developers determine how many stories can be accepted.   **After:**   * Developers meet to break down stories into tasks and assign them (Task Planning). |
| **FAQ/Lesson Learned:** | Scoring of a story should be done quickly - nobody really knows exactly how long it will take. If multiple developers propose different scores, the team should just pick the most voted “score” to get a sense of the complexity. |

## Daily Scrum

*Used to be called Daily Stand-up*

|  |  |
| --- | --- |
| **Problem Statement:** | Things never go as planned, and the development team needs to inspect progress toward the Sprint Goal and adapt the Sprint Backlogs as necessary, adjusting the upcoming planned work to increase their chance of achieving the Sprint Goal. |
| DOs and DONTs**:** | **DOs:**   * Keep it short, ideally not more than 10 - 15 mins; * Inspect progress toward the Sprint Goal; * Synchronize activities to create an actionable plan for the next 24 hours; * Foster focus and self-management.   **DONTs:**   * Spend more than 5 minutes on a particular problem. Remember that the goal is to sync with the rest of the team. For specific issues, you can discuss after everyone shared their update just with the PM and TL. |
| **Attendees:** | Developers (everyone who contributes to the product development), QAs, TL, designer, PM should strive to attend as often as possible (but the Daily Scrum will happen anyway) to efficiently answer any questions that the team might have, and to highlight and promote quick decision-making. The PM should do the best she can to attend at least 3 times a week. |
| **Activities:** | **Before:**   * No specific preparation is required - a brief update should not necessitate preparation.   **During:**   * Share which stories (if any) have been completed; * Share what stories will be started today; * Share all impediments (if any) [e.g. also issues that are slowing down the process without being 100% blockers] * Share any information that may be impacting the sprint goal.   **After:**   * Follow up on the plan that was just shared. |
| **FAQ/Lesson Learned:** | * Hold the meeting at the same time and place to reduce complexity; * Keep it short - this is why it used to be called “stand-up” - it’s not a status meeting. |

## Sprint Retrospective

|  |  |
| --- | --- |
| **Problem Statement:** | The team evaluates the last completed Sprint to identify the most helpful changes to improve quality and effectiveness of process and results. |
| DOs and DONTs**:** | **DOs:**   * Identify the most helpful changes to improve the process and practices to make the next Sprint more effective and enjoyable. * Discuss how the last Sprint went with regards to individuals, interactions, processes, tools, and the Definition of Done. * Focus on top issues rather than all issues (e.g. use Slido to vote top issues).   **DONTs:**   * Spend one hour discussing 20 topics. No meaningful decision can be taken if you have 3 minutes per topic. |
| **Attendees:** | Everyone: Dev, QA, Designer (for UI teams), PM, TL and SM. |
| **Activities:** | **Before:**   * Each team member thinks of possible improvements to the process or to the output.   **During:**   * The team discussed results of actions agreed during the previous Sprint Retrospective (5 mins). * Each member shares their own issues (5 mins). * The team votes on all issues (5 mins). * The team discusses the top issues (40 mins). * The team discusses action items and wraps up the meeting (5 mins).   **After**   * Each member acts on what was discussed and agreed. |
| **FAQ/Lesson Learned:** | * Do not try to discuss too many items. Vote on what most members find important and focus on the top 2-3 issues. * Keep track of past retrospectives (e.g. a new tab in the same Google Sheet) to have some historical perspective (also very good to help new team members getting up to speed). |

## Sprint Review

|  |  |
| --- | --- |
| **Problem Statement:** | The Scrum Team presents the results achieved in the last Sprint to align with and inform stakeholders. |
| **DOs and DONTs:** | **DOs:**   * Share an overview of the sprint: which stories were completed, which stories were not and the expected day for release of completed stories. * Share the most updated view on the team KPIs. * For completed stories, show a live demo as much as possible. * Request all comments to be listed in the Confluence for tracking (the page which details all stories released).   **DONTs:**   * Only show slides. You want, as much as possible, to show the results of the work of the dev team. * Go in details in the reasoning behind the work that was done. The format should focus on problem statements and output. This is a fundamentally different meeting compared to the meetings held BEFORE the Sprint. |
| **Attendees:** | Everyone: Dev, QA, Designer (for UI teams), PM, TL and SM **plus stakeholders**. |
| **Activities:** | **Before:**   * Complete the sprint Confluence update and make sure it’s in sync with the Sprint Review deck (both the “PM page” and the “TL page”). * Get an update on the KPIs. * The PM prepares the deck agenda and discusses with the team on which items she will present and which items will be presented directly by each developer.   **During:**   * Make sure everyone’s feedback is captured in writing (e.g. on the Confluence page) for future actions.   **After:**   * Follow up on action items. * Adapt the product backlog if needed. |
| **FAQ/Lesson Learned:** | * Try as much as possible to show live output rather than only slides. * Engineers are encouraged and welcome to share the results of their work. The PM will present the work otherwise. Note: it’s ok to switch between PM and engineers during the presentation to enable as many engineers as possible to present their own work. |