**Scrum**

Table of Contents

[Steps 4](#_Toc83392999)

[Roles 5](#_Toc83393000)

[Product Owner 5](#_Toc83393001)

[Scrum Master 5](#_Toc83393002)

[Development Team 7](#_Toc83393003)

[Events 8](#_Toc83393004)

[Sprint Planning 8](#_Toc83393005)

[Sprint Review 9](#_Toc83393006)

[Sprint Retrospective 10](#_Toc83393007)

[Daily Scrum 11](#_Toc83393008)

[Artifacts 12](#_Toc83393009)

[Product Backlog 12](#_Toc83393010)

[Sprint Backlog 12](#_Toc83393011)

[Definition of Done 13](#_Toc83393012)

[Burndown Chart 13](#_Toc83393013)

[Story Points 15](#_Toc83393014)

[Story Points vs Time 15](#_Toc83393015)

[Team Capabilities 16](#_Toc83393016)

Scrum is an **agile methodology** that is lightweight and simple to understand, but difficult to master. Like all agile methodologies, the main goal of scrum is not just to get the project completed, but to ensure that the **team is better** at agile at the end of it.

## Steps

In scrum, we have a **product backlog**. This is the list of all the features that need to be implemented in the project. The iterations in scrum are called **sprints**. The length of each sprint is decided beforehand, but is typically 2 to 4 weeks.

At the beginning of each sprint, there is a **planning** stage, where the most important features from the product backlog are selected to be worked on during the sprint. These features are added to the **sprint backlog**. During the sprint, there are also daily meetings, called **scrum meetings**, where shortcomings are discussed.

Once the sprint is complete, there is a **retrospective** stage, where shortcomings during the sprint are discussed to discover ways in which the capability of the team could be increased. At the end of the retrospective stage, **retrospective actions** are taken to implement the improvements. Finally, there is the **review** stage, where the actual product so far is reviewed and potentially released.

## Roles

There are only three roles in a scrum team:

1. Product Owner
2. Scrum Master
3. Development Team

### Product Owner

The **product owner** is responsible for maintaining the **product backlog**. It is their job to ensure requirements are **clearly expressed**, **understood** by each individual to the extent required and **prioritized** to best achieve the goals. Essentially, they **optimize** the value of the work done by the development team.

### Scrum Master

The **scrum master** has several levels of service to perform.

When working with the **product owner**, they must

* Help ensure that the goals, scope and product domain are understood by the everyone
* Find techniques for effective **product backlog management**, for example to decide which software the product owner should use
* Help the team understand the need for clear and concise product backlog items
* Understand and practice **agility**
* Facilitate scrum events

When working with the **development team**, they act as a mentor. They:

* Coach the development team to **self-organize** and be **cross-functional**
* Help them create **high-value products**
* **Remove impediments** to their progress

In terms of the **organization** as a whole, they:

* Lead and coach the organization in **adopting scrum**, perhaps providing training sessions and workshops
* Plan **scrum implementations** within the organization
* Help employees and stakeholders **understand** and **enact scrum**

### Development Team

The **development team** consists of people who work to deliver a potentially releasable product at the end of each sprint.

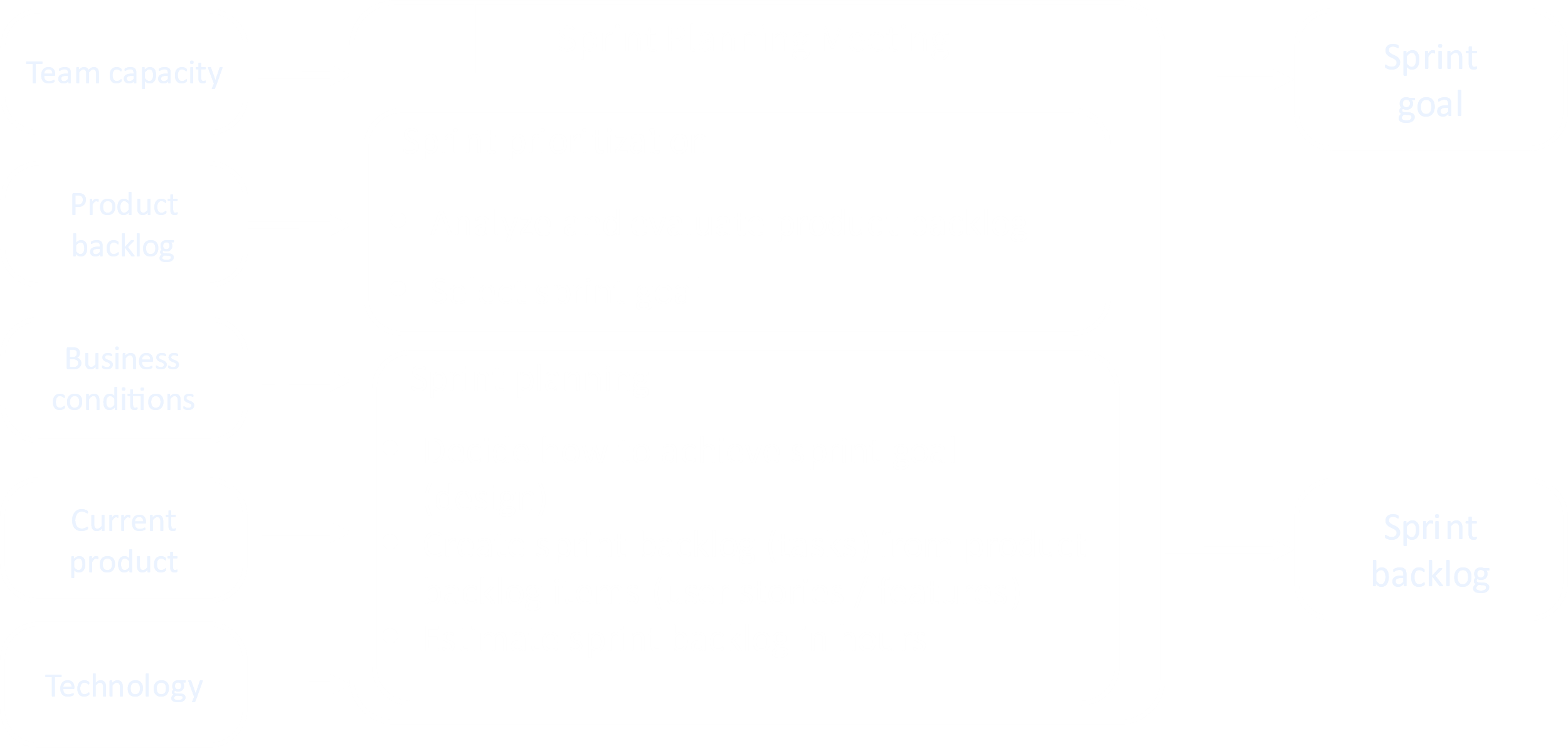
* They are **self-organizing**. Not even the scrum master is allowed to tell them how to do their work or what they will be able to deliver in the next increment. Even if the product owner demands a certain set of features in the next sprint, the development team ultimately decides whether they can do it or not.
* They have no **titles**, regardless of the work an individual does.
* There are no **sub-teams**, even if different domains need to be addressed, like testing, architecture, operations or business analysis.
* Individuals may have **special skills** or areas of focus, but **accountability** falls on the team as a whole.

## Events

### Sprint Planning

**Sprint Planning** is done to answer two questions:

1. What can be delivered in the next sprint?
2. How can it be done?



The **sprint planning meeting** takes the items listed on the left of the above diagram and uses them to answer the two questions.

The product backlog is analysed to prioritize what needs to be done and set the **sprint goal**.

Next, the actual planning stage starts, where it is decided which specific requirements from the product backlog will end up in the **sprint backlog**. We also decide on how to best work on the sprint backlog and how many hours it is estimated to take, although the latter may be quite difficult to determine.

For a **four-week sprint**, the sprint planning meeting may last for **eight hours**. Once the sprint planning meeting ends, **nothing can change** unless specifically demanded by the customer.

### Sprint Review

At the end of each sprint, there is a **sprint review meeting**. The scrum team and key stakeholders are asked to attend by the **product owner**. The purpose of the meeting is:

* The list of **product backlog** items that are complete and incomplete is examined. This includes items that were meant to be completed in the previous sprint but could not be.
* The **development team** discusses what went well during the sprint and what did not. For the issues faced, solutions are discussed.
* The entire group discusses **what to do next**. This involves taking into account how the market may have changed since the last sprint.
* The timeline, budget, potential capabilities and the marketplace are reviewed for the next **anticipated release** of the product.

The sprint review meeting tends to last **two to four hours**.

### Sprint Retrospective

The purpose of the **sprint retrospective** is to:

* Inspect how the last sprint went with regards to people, relationships, processes, and tools
* Identify and order the major items that went well and potential improvements and
* Create a plan for implementing improvements to the way the Scrum Team does its work

Essentially, the whole team gathers to discuss what they want to start doing, what they want to stop doing and what they want to continue doing.

The above is a nice way of putting it. In practice, **people fight**. They complain about the things that went badly, even about each other. In this way, all the issues faced come out and ways to resolve them can be discussed.

Sprint retrospectives also take place after the sprint and last for **two to four hours**.

### Daily Scrum

The most important meeting in this list is the **daily scrum**. It happens **every day**, lasts **15 minutes** and everyone has to be **standing**.

The daily scrum is not for **problem solving** in the way that the other meetings are. In the daily scrum, the scrum team discusses what they did yesterday, what they will do today and if they are facing any problems. Any problems mentioned will not be solved during the meeting, but later on, by the **scrum master**.

The only people who are obligated to attend a daily scrum are the members of the **development team**. Anyone else is allowed to come to a daily scrum, but only members from the development team, the product owner or the scrum master are allowed to talk.

## Artifacts

### Product Backlog

The **product backlog** is the list of all desired work in the project, or its **requirements**. Ideally, the requirements are expressed in terms of their value to users. The product backlog is **prioritized** by the **product owner** and **reprioritized** at the start of each **sprint**.

### Sprint Backlog

The **sprint backlog** is the set of product backlog items selected for the **current sprint**. It is a prediction of what will be made available by the **development team** in the **next increment**.

The sprint backlog is not supposed to change, but it can change for one of two reasons:

1. Over time, the development team will gain a **better understanding** of the requirements. At that point, they may find that new tasks need to be added to the sprint backlog.
2. **Bugs** of existing features must be fixed. These are added as new tasks to the sprint backlog.

### Definition of Done

For every requirement, the goal is to get it ‘done’. A particular item is declared to be **done** when its **acceptance criteria** are met. This may include quality criteria, constraints and non-functional requirements.

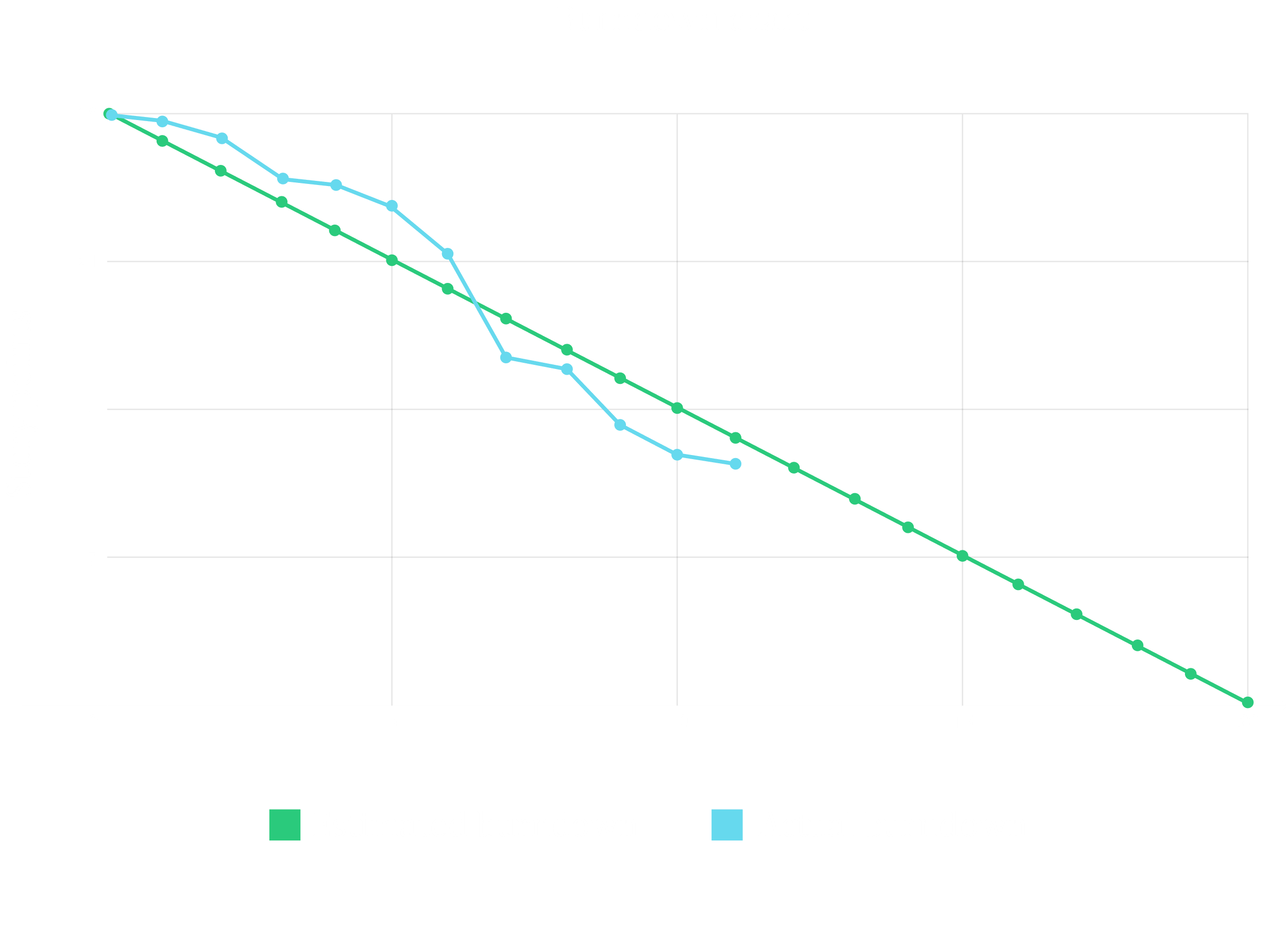
The following checklist is useful when determining if a particular requirement can be declared done:

* Reviewed by someone or a particular stakeholder
* Completed unit acceptance testing of the User Story
* Completion of quality assurance tests
* Completion of all documentation related to the User Story
* All issues are fixed
* Successful demonstration to stakeholders and/or business representatives

### Burndown Chart

A **burndown chart** is a graph that shows an **overview of progress** over time. It starts with the total number of **story points** taken on for the current sprint and tracks how many of those points are completed on a **daily basis**. As tasks are completed, the graph ‘burns down’ to . A typical chart additionally shows the ‘ideal’ burndown rate, which is just a constant rate.

The burndown chart is usually maintained by the **scrum master** and update daily, usually after the daily scrum.

****

## Story Points

User requirements are usually expressed as **user stories**. Each story gives us some requirement. Each story is assigned a story point, which is essentially used to express how **difficult** it will be to implement. Before assigning the story point the **complexity** of the requirement, the **effort** required to implement it and the number of **unknown factors**, essentially the risk, are taken into consideration.

Story points can be given values in two ways:

1. Following a Fibonacci sequence, i.e. 1, 2, 3, 5, 8, 13, 21, 34
2. Following T-Shirt sizes, i.e. Small: 1 – 3, Medium: 5 – 8, Large: 13 – 20, Extra Large: 40+

### Story Points vs Time

An argument could be that **time** be used to describe complexity instead of story points. For example, we could say that a particular requirement takes weeks to implement. This is after all, easier to understand.

The issue here is that the time required will **vary** depending on the developer assigned. One developer may be less skilled and thus require more time to complete a task and vice versa. Story points act as a measure of difficulty that is **independent** of the person performing the task.

### Team Capabilities

Say we have a hundred user stories in the product backlog. From here, we need to take a few into the sprint backlog. When doing this, we need to take into account the **capabilities** of the development team. In some sprints, we may find that the team completed more story points than was originally planned, while in others, the opposite may happen. Inspecting these variations can help determine the right amount of story points to assign to the team in each sprint. This could just be the **average** of the story points completed in the previous sprints.