**SCRUM**

**https://www.mountaingoatsoftware.com/agile**

**WHAT IS SCRUM?**

**Scrum is an agile way to manage a project, usually software development.**

Agile software development with Scrum is often perceived as a methodology; but rather than viewing Scrum as methodology, think of **it as a framework for managing a process**.

In the agile Scrum world, instead of providing complete, detailed descriptions of how everything is to be done on a project, much of it is left up to the Scrum software development team. This is because the team will know best how to solve the problem they are presented.

This is why in Scrum development, for example, a **sprint planning meeting** is described in terms of the desired outcome (a commitment to a set of features to be developed in the next sprint) instead of a set of Entry criteria, Task definitions, Validation criteria, Exit criteria (ETVX) and so on, as would be provided in most methodologies.

**Scrum relies on a self-organizing, cross-functional team.** The scrum team is self-organizing in that there is no overall team leader who decides which person will do which task or how a problem will be solved. Those are issues that are decided by the team as a whole.

And in Scrum, a team is cross functional, meaning everyone is needed to take a feature from idea to implementation.

Within agile development, Scrum teams are supported by two specific roles. The first is a **Scrum Master**, who can be thought of as a coach for the team, helping team members use the Scrum process to perform at the highest level.

The **product owner (PO)** is the other role, and in Scrum software development, represents the business, customers or users, and guides the team toward building the right product.

**Scrum Development: What’s involved?**

The Scrum model suggests that projects progress via a series of **sprints**. In keeping with an agile methodology, sprints are timeboxed to no more than a month long, most commonly two weeks.

Scrum methodology advocates for a planning meeting at the start of the sprint, where team members figure out how many items they can commit to, and then create a **sprint backlog** – a list of the tasks to perform during the sprint.

During an agile Scrum sprint, the Scrum team takes a small set of features from idea to coded and tested functionality. At the end, these features are done, meaning coded, tested and integrated into the evolving product or system.

On each day of the sprint, all team members should attend a **daily Scrum meeting**, including the Scrum Master and the product owner. This meeting is timeboxed to no more than 15 minutes. During that time, team members share what they worked on the prior day, will work on that day, and identify any impediments to progress.

The Scrum model sees daily scrums as a way to synchronize the work of team members as they discuss the work of the sprint.

At the end of a sprint, the team conducts a **sprint review meeting** during which the team demonstrates the new functionality to the PO or any other stakeholder who wishes to provide feedback that could influence the next sprint.

This feedback loop within Scrum software development may result in changes to the freshly delivered functionality, but it may just as likely result in revising or adding items to the product backlog.

Another activity in Scrum project management is the **sprint retrospective** at the end of each sprint. The whole team participates in this meeting, including the Scrum Master and PO. The meeting is an opportunity to reflect on the sprint that has ended, and identify opportunities to improve.

**Scrum Process: The Main Artifacts**

The primary artifact in Scrum development is, of course, the **product** itself. The Scrum model expects the team to bring the product or system to a potentially shippable state at the end of each Scrum sprint.

The **product backlog** is another artifact of Scrum. This is the complete list of the functionality that remains to be added to the product. The product owner prioritizes the backlog so the team always works on the most valuable features first.

The most popular and successful way to create a product backlog using Scrum methodology is to populate it with **user stories**, which are short descriptions of functionality described from the perspective of a user or customer.

In Scrum project management, on the first day of a sprint and during the planning meeting, team members create the **sprint backlog**. The sprint backlog can be thought of as the team's to-do list for the sprint, whereas a product backlog is a list of features to be built (written in the form of user stories).

The sprint backlog is the list of tasks the team needs to perform in order to deliver the functionality it committed to deliver during the sprint.

Additional artifacts resulting from the Scrum agile methodology is the **sprint burndown chart** and **release burndown chart**. Burndown charts show the amount of work remaining either in a sprint or a release, and are an effective tool in Scrum software development to determine whether a sprint or release is on schedule to have all planned work finished by the desired date.

**The Agile Scrum Project: Main Roles**

Even if you are new to Scrum, you may have heard of a role called the ScrumMaster. The ScrumMaster is the team's coach, and helps Scrum practitioners achieve their highest level of performance.

In the Scrum process, a **ScrumMaster** differs from a traditional project manager in many ways, including that this role does not provide day-to-day direction to the team and does not assign tasks to individuals.

A good ScrumMaster shelters the team from outside distractions, allowing team members to focus maniacally during the sprint on the goal they have selected.

While the ScrumMaster focuses on helping the team be the best that it can be, the **product owner** works to direct the team to the right goal. The product owner does this by creating a compelling vision of the product, and then conveying that vision to the team through the product backlog.

The product owner is responsible for prioritizing the backlog during Scrum development, to ensure it’s up to par as more is learned about the system being built, its users, the team and so on.

The third and final role in Scrum project management is the **Scrum team** itself. Although individuals may join the team with various job titles, in Scrum, those titles are insignificant. Scrum methodology states that each person contributes in whatever way they can to complete the work of each sprint.

This does not mean that a tester will be expected to re-architect the system; individuals will spend most (and sometimes all) of their time working in whatever discipline they worked before adopting the agile Scrum model. But with Scrum, individuals are expected to work beyond their preferred disciplines whenever doing so would be for the good of the team.

One way to think of the interlocking nature of these three roles in this agile methodology is as a race car.

The Scrum team is the car itself, ready to speed along in whatever direction it is pointed. The product owner is the driver, making sure that the car is always going in the right direction. And the ScrumMaster is the chief mechanic, keeping the car well-tuned and performing at its best.

**Scrum Overview for Agile Software Development**

[Scrum](https://www.mountaingoatsoftware.com/agile/scrum) is an agile process most commonly used for product development, especially software development. Scrum is a project management framework that is [applicable to any project](http://blog.mountaingoatsoftware.com/deciding-what-kind-of-projects-are-most-suited-for-agile) with aggressive deadlines, complex requirements and a degree of uniqueness. In Scrum, projects move forward via a series of iterations called sprints. Each sprint is typically two to four weeks long.

**Scrum Overview - Introduction to Scrum Terms**

An introduction to Scrum would not be complete without knowing the Scrum terms you'll be using. This section in the Scrum overview will discuss common concepts in Scrum.

**Scrum team:** A typical scrum team has between five and nine people, but Scrum projects can easily scale into the hundreds. However, Scrum can easily be used by one-person teams and often is. This team does not include any of the traditional software engineering roles such as programmer, designer, tester or architect. Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint. Scrum teams develop a deep form of camaraderie and a feeling that “we’re all in this together.”

**Product owner:** The product owner is the project’s key stakeholder and represents users, customers and others in the process. The product owner is often someone from product management or marketing, a key stakeholder or a key user.

**ScrumMaster:** The ScrumMaster is responsible for making sure the team is as productive as possible. The ScrumMaster does this by helping the team use the Scrum process, by removing impediments to progress, by protecting the team from outside, and so on.

**Product backlog:** The product backlog is a prioritized features list containing every desired feature or change to the product. Note: The term “backlog” can get confusing because it’s used for two different things. To clarify, the product backlog is a list of desired features for the product. The sprint backlog is a list of tasks to be completed in a sprint.

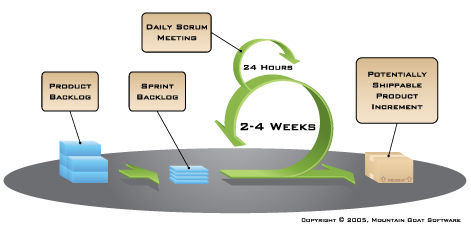
**Sprint planning meeting:** At the start of each sprint, a sprint planning meeting is held, during which the product owner presents the top items on the product backlog to the team. The Scrum team selects the work they can complete during the coming sprint. That work is then moved from the product backlog to a sprint backlog, which is the list of tasks needed to complete the product backlog items the team has committed to complete in the sprint.

**Daily Scrum:** Each day during the sprint, a brief meeting called the daily scrum is conducted. This meeting helps set the context for each day’s work and helps the team stay on track. All team members are required to attend the daily scrum.

**Sprint review meeting:** At the end of each sprint, the team demonstrates the completed functionality at a sprint review meeting, during which, the team shows what they accomplished during the sprint. Typically, this takes the form of a demonstration of the new features, but in an informal way; for example, PowerPoint slides are not allowed. The meeting must not become a task in itself nor a distraction from the process.

**Sprint retrospective:** Also at the end of each sprint, the team conducts a sprint retrospective, which is a meeting during which the team (including its ScrumMaster and product owner) reflect on how well Scrum is working for them and what changes they may wish to make for it to work even better.

**A Visual Introduction to Scrum**



This graphic is an introduction to the essential elements of using Scrum for agile software development. On the left, we see the product backlog, which has been prioritized by the product owner and contains everything desired in the product that’s known at the time. The two to four week sprints are shown by the larger green circle.

At the start of each sprint, the team selects some amount of work from the product backlog and commits to completing that work during the sprint. Part of figuring out how much they can commit to is creating the sprint backlog, which is the list of tasks (and an estimate of how long each will take) needed to deliver the selected set of product backlog items to be completed in the sprint.

At the end of each sprint, the team produces a potentially shippable product increment — i.e. working, high-quality software. Each day during the sprint, team members meet to discuss their progress and any impediments to completing the work for that sprint. This is known as the daily scrum, and is shown as the smaller green circle above.

**Introduction to Scrum Agile Trainer, Coach and Consultant Mike Cohn**

Mountain Goat Software founder Mike Cohn is a Scrum and agile specialist. Along with Scrum inventor Ken Schwaber, Mike was a co-founder of the non-profit Scrum Alliance, which is dedicated to worldwide support for Scrum and Scrum practitioners. He is also a co-founder of the non-profit Agile Alliance. Having run his first Scrum project in 1995, Mike is among the most experienced of Scrum trainers and coaches. To learn more about Mountain Goat Software and Mike Cohn, check out this page: [Why Mountain Goat?](https://www.mountaingoatsoftware.com/company/why-us)

**ScrumMaster**

What is a Scrum Master? The ScrumMaster is responsible for making sure a Scrum team lives by the values and practices of [Scrum](https://www.mountaingoatsoftware.com/agile/scrum). The ScrumMaster is often considered a coach for the team, helping the team do the best work it possibly can. The ScrumMaster can also be thought of as a *process owner* for the team, creating a balance with the project's key stakeholder, who is referred to as the product owner.

The ScrumMaster does anything possible to help the team perform at their highest level. This involves removing any impediments to progress, facilitating meetings, and doing things like working with the product owner to make sure the product backlog is in good shape and ready for the next sprint. The ScrumMaster role is commonly filled by a former project manager or a technical team leader but can be anyone.

The ScrumMaster is also often viewed as a protector of the team. The most common example is that the ScrumMaster protects the team by making sure they do not over-commit themselves to what they can achieve during a sprint due to pressure from an overly aggressive product owner. However, a good ScrumMaster also [protects the team from complacency](http://blog.mountaingoatsoftware.com/protecting-the-team-cuts-both-ways).

(complacency meaning : [self-satisfaction](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-satisfaction&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIHTAA), [self-approval](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-approval&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIHjAA), [self-approbation](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-approbation&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIHzAA), [self-admiration](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-admiration&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIIDAA), [self-congratulation](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-congratulation&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIITAA), [self-regard](https://www.google.co.in/search?rlz=1C1CHBD_enIN691IN691&espv=2&biw=1920&bih=955&site=webhp&q=define+self-regard&sa=X&ved=0ahUKEwjb3PKE-bbNAhWMu48KHQ2sB5IQ_SoIIjAA))

What is a Scrum Master role and how does it fit into the project? Many who are new to the ScrumMaster role struggle with the apparent contradiction of the ScrumMaster as both **a servant-leader** to the team and also someone with no authority? The seeming contradiction disappears when we realize that although the ScrumMaster has no authority over Scrum team members, the ScrumMaster does have authority over the process. Although a ScrumMaster may not be able to say, “You’re fired,” a ScrumMaster can say, “I’ve decided we’re going to try two-week sprints for the next month.”

The ScrumMaster is there to help the team in its use of Scrum. Think of the help from a ScrumMaster as similar to a personal trainer who helps you stick with an exercise regimen and perform all exercises with the correct form. A good trainer will provide motivation while at the same time making sure you don’t cheat by skipping a hard exercise. The trainer’s authority, however, is limited. The trainer cannot make you do an exercise you don’t want to do. Instead, the trainer reminds you of your goals and how you’ve chosen to meet them. To the extent that the trainer does have authority, it has been granted by the client. ScrumMasters are much the same: They have authority, but that authority is granted to them by the team.

A ScrumMaster can say to a team, “Look, we’re supposed to deliver potentially shippable software at the end of each sprint. We didn’t do that this time. What can we do to make sure we do better the next sprint?” This is the ScrumMaster exerting authority over the process; something has gone wrong with the process if the team has failed to deliver something potentially shippable.

But because the ScrumMaster’s authority does not extend beyond the process, the same ScrumMaster should not say, “Because we failed to deliver something potentially shippable the last sprint, I want Tod to review all code before it gets checked in.” Having Tod review the code might be a good idea, but the decision is not the ScrumMaster’s to make. Doing so goes beyond authority over the process and enters into how the team works.

With authority limited to ensuring the team follows the process, the ScrumMaster’s role can be more difficult than that of a typical project manager. Project managers often have the fallback position of “do it because I say so.” The times when a ScrumMaster can say that are limited and restricted to ensuring that Scrum is being followed.

Read more about the [role of the ScrumMaster](http://blog.mountaingoatsoftware.com/tag/scrummaster), [ScrumMaster requirements](http://www.mountaingoatsoftware.com/agile/scrum/scrummaster/requirements/)and the [six attributes of the ideal ScrumMaster](https://www.mountaingoatsoftware.com/articles/34-leader-of-the-band).

**Sprint Planning Meeting**

In [Scrum](https://www.mountaingoatsoftware.com/agile/scrum), the sprint planning meeting is attended by the product owner, ScrumMaster and the entire Scrum team. Outside stakeholders may attend by invitation of the team, although this is rare in most companies.

During the sprint planning meeting, the product owner describes the highest priority features to the team. The team asks enough questions that they can turn a high-level user story of the product backlog into the more detailed tasks of the sprint backlog.

The product owner doesn't have to describe every item being tracked on the product backlog. A good guideline is for the product owner to come to the sprint planning meeting prepared to talk about two sprint's worth of product backlog items. To make an example really simple, suppose a team always finishes five product backlog items. Their product owner should enter the meeting prepared to talk about the top 10 priorities.

There are two defined artifacts that result from a sprint planning meeting:

A **sprint goal**

A **sprint backlog**

A sprint goal is a short, one- or two-sentence, description of what the team plans to achieve during the sprint. It is written collaboratively by the team and the product owner. The following are example sprint goals on an eCommerce application:

Implement basic shopping cart functionality including add, remove, and update quantities.

Develop the checkout process: pay for an order, pick shipping, order gift wrapping, etc.

The sprint goal can be used for quick reporting to those outside the sprint. There are always stakeholders who want to know what the team is working on, but who do not need to hear about each product backlog item (user story) in detail. The success of the sprint will later be assessed during the sprint review meeting against the sprint goal, rather than against each specific item selected from the product backlog.

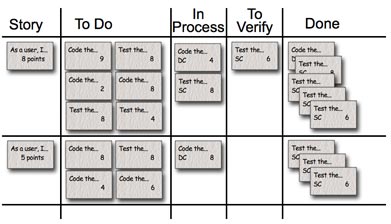
The sprint backlog is the other output of sprint planning. A sprint backlog is a list of the product backlog items the team commits to delivering *plus* the list of tasks necessary to delivering those product backlog items. Each task on the sprint backlog is also usually estimated.

An important point to reiterate here is that it's the team that selects how much work they can do in the coming sprint. The product owner does not get to say, "We have four sprints left so you need to do one-fourth of everything I need." We can hope the team does that much (or more), but it's up to the team to determine how much they can do in the sprint.

**Scrum Task Board**

When practicing [Scrum](https://www.mountaingoatsoftware.com/agile/scrum), we can make the sprint backlog visible by putting it on aScrum task board. Team members update the task board continuously throughout the sprint; if someone thinks of a new task (“Test the snark code on Windows 8.1”), she writes a new card and puts it on the wall. Either during or before the daily scrum, estimates are changed (up or down), and cards are moved around the board.

As an example, the Scrumboard looks like this:

[](https://www.mountaingoatsoftware.com/uploads/articles/MockedTaskBoard.jpg)

(A generic taskboard.)

Each row on the Scrum board is a user story, which is the unit of work we encourage teams to use for their product backlog. During the sprint planning meeting, the team selects the product backlog items they can complete during the coming sprint. Each product backlog item is turned into multiple sprint backlog items. Each of these is represented by one task card that is placed on the Scrum board. Each task card starts on the Scrum task board in the “To Do” column. The columns we generally use on a taskboard are:

**Story:** The story description (“As a user we want to…”) shown on that row.

**To Do:** Place for all cards that are not in the “Done” or “In Process” columns for the current sprint.

**Work In Process:** Any card being worked on goes here. The programmer who chooses to work on it moves it over when she's ready to start the task. Often, this happens during the daily scrum when someone says, “I'm going to work on the boojum today.”

**To Verify:** A lot of tasks have corresponding test task cards. So, if there's a “Code the boojum class” card, there is likely one or more task cards related to testing: “Test the boojum”, “Write FitNesse tests for the boojum,” “Write FitNesse fixture for the boojum,” etc. Some task cards don't get corresponding test cards (“Fix Bug No. 321 in Bugzilla”) so those are placed in the “To Verify” column.

**Done:** Cards pile up over here when they're done. They're removed at the end of the sprint. Sometimes we remove some or all during a sprint if there are a lot of cards.

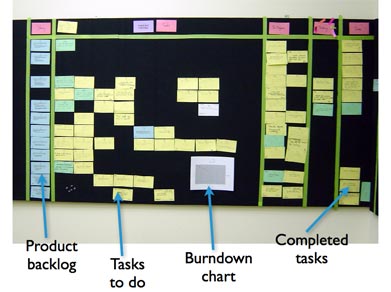
Optionally, we sometimes use the following columns on a Scrum task board, depending on the team, the culture, the project and other considerations:

**Notes:** Just a place to jot a note or two.

**Tests Specified:** We like to do “Story Test-Driven Development,” or “Acceptance Test-Driven Development,” which means the tests are written before the story is coded. Many teams find that it helps to have acceptance tests identified before coding begins on a particular story. This column just contains a checkmark to indicate the tests are specified.

Here are some photos of actual task boards in use. Click on any to enlarge.

A Scrumboard hanging in a team room:

[](https://www.mountaingoatsoftware.com/uploads/articles/LabelledTaskBoard.jpg)

Cork board hung on the wall:

[](https://www.mountaingoatsoftware.com/uploads/articles/CorkTaskBoard.jpg)

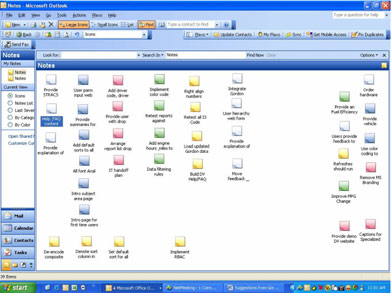
A metal taskboard with cards placed with magnets:

[](https://www.mountaingoatsoftware.com/uploads/articles/MagneticTaskBoard.jpg)

A Scrum task board made with black tape on a large wall-sized cabinet (there's food in the cabinet!):

[](https://www.mountaingoatsoftware.com/uploads/articles/CabinetTaskBoard.jpg)

A distributed team using Outlook's notes facility on a shared desktop:

[](https://www.mountaingoatsoftware.com/uploads/articles/SharedDesktopTaskBoard.jpg)

For more resources on Scrum task boards, check out these blog posts:

[Using a Task Board with One Remote Team Member](https://www.mountaingoatsoftware.com/blog/using-a-task-board-with-one-remote-team-member)

[Working with Storyless Tasks](https://www.mountaingoatsoftware.com/blog/working-with-storyless-tasks)

[The Ideal Agile Workspace](https://www.mountaingoatsoftware.com/blog/the-ideal-agile-workspace)

**The Chicken and the Pig**



There is a story in Scrum about a chicken and a pig.

One day the chicken decides that the two should start a restaurant.

The pig is intrigued by the idea and says, “That sounds great. I’m an entrepreneurial type of hog. I’m sick of working for the farmer. But what are we going to call the restaurant?”

The chicken thinks. Then she scratches and pecks at the dirt and suggests, “Ham and Eggs!”

To which the pig replies, “No thanks, I’d be committed. You’d only be involved.”

The Difference between **Commitment and Involvement**

This story is told in Scrum circles to point out the difference between commitment and involvement.

The original idea was that the development team members and Scrum Master were committed, but the product owner was merely involved. It was used to claim that the product owner should not participate (or even attend) the daily scrum.

I’ve never been fond of that way of thinking. It sets up an unnecessary divide between team and product owner. Further, who can be more committed to the success of a project than the product owner, who is sometimes called the “single wringable neck” in Scrum literature?

**Daily Scrum Meeting**

In [Scrum](https://www.mountaingoatsoftware.com/agile/scrum), on each day of a sprint, the team holds a daily scrum meeting called the "daily scrum.” Meetings are typically held in the same location and at the same time each day. Ideally, a daily scrum meeting is held in the morning, as it helps set the context for the coming day's work. These scrum meetings are strictly time-boxed to 15 minutes. This keeps the discussion brisk but relevant.

There is an old joke in Scrum about a chicken and a pig that [illustrates the differences](https://www.mountaingoatsoftware.com/agile/scrum/the-chicken-and-the-pig) between being committed and being involved.

Scrum affords special status to those who are committed, and many teams enforce a rule in which only those who are committed are allowed to talk during the daily scrum meeting.

The team and Scrum Master are considered committed by nearly everyone in the Scrum community. There is some disagreement about the product owner. My view is that a product owner should be considered a dedicated participant of the project. (And should behave as one, too.)

All team members are required to attend scrum meetings. Since both the ScrumMaster and product owner are committed team members, they are expected to attend and participate. Anyone else (for example, a departmental VP, a salesperson or a developer from another project) is allowed to attend, but is there only to listen. This makes scrum meetings an excellent way for a Scrum team to disseminate information -- if you're interested in hearing where things are at, attend that day's meeting.

The daily scrum meeting is not used as a problem-solving or issue resolution meeting. Issues that are raised are taken offline and usually dealt with by the relevant subgroup immediately after the meeting. During the daily scrum, each team member answers the following three questions:

What did you do yesterday?

What will you do today?

Are there any impediments in your way?

By focusing on what each person accomplished yesterday and will accomplish today, the team gains an excellent understanding of what work has been done and what work remains. The daily scrum meeting is not a status update meeting in which a boss is collecting information about who is behind schedule. Rather, it is a meeting in which team members make commitments to each other.

If a programmer stands up and says, "Today, I will finish the data storage module," everyone knows that in tomorrow's meeting, he will say whether or not he finished. This has the wonderful effect of helping a team realize the significance of these commitments, and that their commitments are to*one another,* not to some far-off customer or salesman.

Any impediments that are raised in the scrum meeting become the ScrumMaster's responsibility to resolve as quickly as possible. Typical impediments are:

My \_\_\_\_ broke and I need a new one today.

I still haven't got the software I ordered a month ago.

I need help debugging a problem with \_\_\_\_\_\_.

I'm struggling to learn \_\_\_\_\_\_ and would like to pair with someone on it.

I can't get the vendor's tech support group to call me back.

Our new contractor can't start because no one is here to sign her contract.

I can't get the \_\_\_\_ group to give me any time and I need to meet with them.

The department VP has asked me to work on something else "for a day or two."

In cases where the ScrumMaster cannot remove these impediments directly himself (e.g., usually the more technical issues), he still takes responsibility for making sure someone on the team does quickly resolve the issue.

The vast majority of teams conduct the daily scrum meeting by having each person answer the three questions in order. You answer all three, then the next person, the next and so on. An interesting alternative that some teams find helpful is to [talk through one product backlog item](http://blog.mountaingoatsoftware.com/should-the-daily-standup-be-person-by-person-or-story-by-story) before moving on to the next. In this way, an individual may give an update at multiple different times during the same meeting.

For more information on daily scrum meetings, check out these blog posts:

[Daily Scrum: Not Just for ScrumMasters](https://www.mountaingoatsoftware.com/blog/daily-scrum-not-just-for-scrummasters)

[A Weighty Matter for Daily Scrum](http://www.mountaingoatsoftware.com/blog/weighty-matter-daily-scrum)

**Product Owner**

The [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) product owner is typically a project's key stakeholder. Part of the product owner responsibilities is to have a vision of what he or she wishes to build, and convey that vision to the scrum team. This is key to successfully starting any agile software development project. The agile product owner does this in part through the product backlog, which is a prioritized features list for the product.

The product owner is commonly a lead user of the system or someone from marketing, product management or anyone with a solid understanding of users, the market place, and the competition and of future trends for the domain or type of system being developed.

This, of course, varies tremendously based on whether the team is developing commercial software, software for internal use, hardware or some other type of product. The key is that the person in the product owner role needs to have a vision for what is to be built.

Although the agile PO prioritizes the product backlog during the sprint planning meeting, the team selects the amount of work they believe they can do during each sprint, and how many sprints will be required.   
  
The product owner does not get to say, "We have four sprints left, therefore you must do one-fourth of the product backlog this sprint." The Scrum product owner's job is to motivate the team with a clear, elevating goal. Team members know best what they are capable of, and so they select which user stories from the top of the product backlog they can commit to delivering during any sprint.

In return for the Scrum team's commitment to completing the selected user stories from the top of the product backlog, the product owner makes a reciprocal commitment to not throw new requirements at the team during the sprint. Requirements are allowed to change (and change is encouraged) but only outside the sprint. Once the team starts on a sprint, it remains maniacally focused on the goal of that sprint.  
  
The product owner role requires an individual with certain skills and traits, including availability, business savvy and communication skills. First, the Scrum product owner needs to be available to his or her team. The best product owners show commitment by doing whatever is necessary to build the best product possible – and that means being actively engaged with their teams.  
  
Business savvy is important for the agile product owner because he or she is the decision maker regarding what features the product will have. That means, the agile PO should understand the market, the customer and the business in order to make sound decisions.

Finally, communication is a large part of the product owner responsibilities. The product owner role requires working closely with key stakeholders throughout the organization and beyond, so he or she must be able to communicate different messages to different people about the project at any given time.

**Scrum Team**

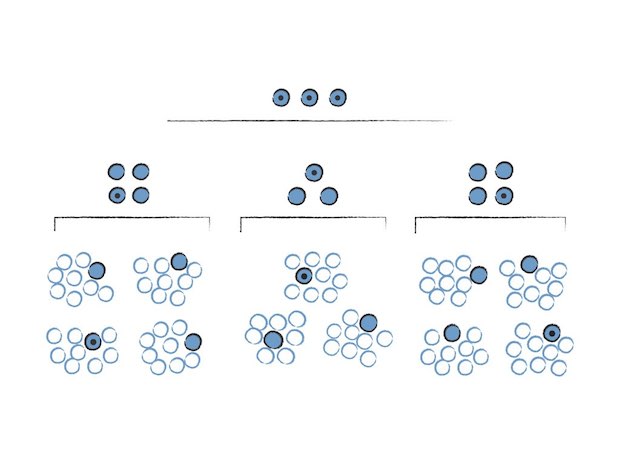
A Scrum team in a [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) environment does not include any of the traditional software engineering roles such as programmer, designer, tester or architect. Everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint. Because of this, Scrum teams develop a deep form of camaraderie and a feeling that "we're all in this together."

Former roles in traditional teams often adapt to an agile role that makes them an integral Scrum team member who retains some of the aspects of their prior role, but also adds new traits as well. New roles in a Scrum team are the ScrumMaster or product owner. You can find out more about which role in a Scrum team would suit you [here](http://www.mountaingoatsoftware.com/training/roles) by finding your pre-agile role.

A typical Scrum team is five to nine people. Rather than scaling by having a large team, Scrum projects scale through having teams of teams. In this way, we have worked on projects with more than 500 people and have consulted on projects with more than 1,000.

Although it's not the only thing necessary to scale Scrum, one well-known technique is the use of a **"Scrum of Scrums" meeting**. With this approach, each Scrum team proceeds as normal, but each team identifies one person who attends the Scrum of Scrums meeting to coordinate the work of multiple Scrum teams. These meetings are analogous to the daily Scrum meeting, but do not necessarily happen every day. In many organizations, having a Scrum of Scrums meeting two or three times a week is sufficient.

The illustration below shows how a Scrum of Scrums approach allows Scrum to scale up (in this case to 243 people). Each cell represents one person on a Scrum team. The bottom of this illustration shows teams with nine developers on them. One person from each team (the differently colored cell) also participates in a Scrum of Scrum to coordinate work above that team. Then from those nine-person teams, another person is selected (this time shown with diagonal lines) to participate in what is called a Scrum of Scrums of Scrums.



**Sprint Retrospective**

No matter how good a [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) team is, there is always opportunity to improve. Although a good Scrum team will be constantly looking for improvement opportunities, the team should set aside a brief, dedicated period at the end of each sprint to deliberately reflect on how they are doing and to find ways to improve. This occurs during the sprint retrospective.

The sprint retrospective is usually the last thing done in a sprint. Many teams will do it immediately after the sprint review. The entire team, including both the ScrumMaster and the product owner should participate. You can schedule a scrum retrospective for up to an hour, which is usually quite sufficient. However, occasionally a hot topic will arise or a team conflict will escalate and the retrospective could take significantly longer.

Although there are many [ways to conduct an agile sprint retrospective](http://www.mountaingoatsoftware.com/reviews/27-agile-retrospectives), our recommendation is to conduct it as a start-stop-continue meeting. This is perhaps the simplest, but often the most effective way to conduct a retrospective. Using this approach each team member is asked to identify specific things that the team should:

Start doing

Stop doing

Continue doing

There are many variations on this simple format. The ScrumMaster can facilitate this sprint retrospective meeting by asking everyone to just shout out ideas during the scrum. The ScrumMaster can go around the room asking each person to identify any one thing to start, stop or continue. Or, for example, he or she can tell everyone to focus on identifying something to stop this time because not much attention has been paid to things to stop in recent retrospectives.  
  
After an initial list of ideas has been brainstormed, teams will commonly vote on specific items to focus on during the coming sprint. At the end of the sprint, the next retrospective is often begun by reviewing the list of things selected for attention in the prior sprint retrospective.



**Scrum Product Backlog**

The agile product backlog in [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) is a prioritized features list, containing short descriptions of all functionality desired in the product. When applying Scrum, it's not necessary to start a project with a lengthy, upfront effort to document all requirements. Typically, a Scrum team and its product owner begin by writing down everything they can think of for agile backlog prioritization. This agile product backlog is almost always more than enough for a first sprint. The Scrum product backlog is then allowed to grow and change as more is learned about the product and its customers.

A typical Scrum backlog comprises the following different types of items:

Features

Bugs

Technical work

Knowledge acquisition

By far, the predominant way for a Scrum team to express features on the agile product backlog is in the form of user stories, which are short, simple descriptions of the desired functionality told from perspective of the user. An example would be, "As a shopper, I can review the items in my shopping cart before checking out so that I can see what I've already selected."

Because there's really no difference between a bug and a new feature -- each describes something different that a user wants -- [bugs are also put](http://blog.mountaingoatsoftware.com/bugs-on-the-product-backlog) on the Scrum product backlog.

Technical work and knowledge acquisition activities also belong on the agile backlog. An example of technical work would be, "Upgrade all developers' workstations to Windows 7." An example of knowledge acquisition could be a Scrum backlog item about researching various JavaScript libraries and making a selection.  
  
The product owner shows up at the sprint planning meeting with the prioritized agile product backlog and describes the top items to the team. The team then determines which items they can complete during the coming sprint. The team then moves items from the product backlog to the sprint backlog. In doing so, they expand each Scrum product backlog item into one or more sprint backlog tasks so they can more effectively share work during the sprint.  
  
Conceptually, the team starts at the top of the prioritized Scrum backlog and draws a line after the lowest of the high-priority items they feel they can complete. In practice, it's not unusual to see a team select, for example, the top five items and then two items from lower on the list that are associated with the initial five.

For more resources on this topic, check out these blog posts:

[Sample Format for a Spreadsheet Product Backlog](https://www.mountaingoatsoftware.com/blog/a-sample-format-for-a-spreadsheet-based-product-backlog)

[GASPing About the Product Backlog](https://www.mountaingoatsoftware.com/blog/gasping-about-the-product-backlog)

[Mix the Size of Product Backlogs You Committ To](https://www.mountaingoatsoftware.com/blog/mix-the-sizes-of-the-product-backlog-items-you-commit-to)

[Visualizing a Large Product Backlog with a Treemap](https://www.mountaingoatsoftware.com/blog/visualizing-a-large-product-backlog-with-a-treemap)

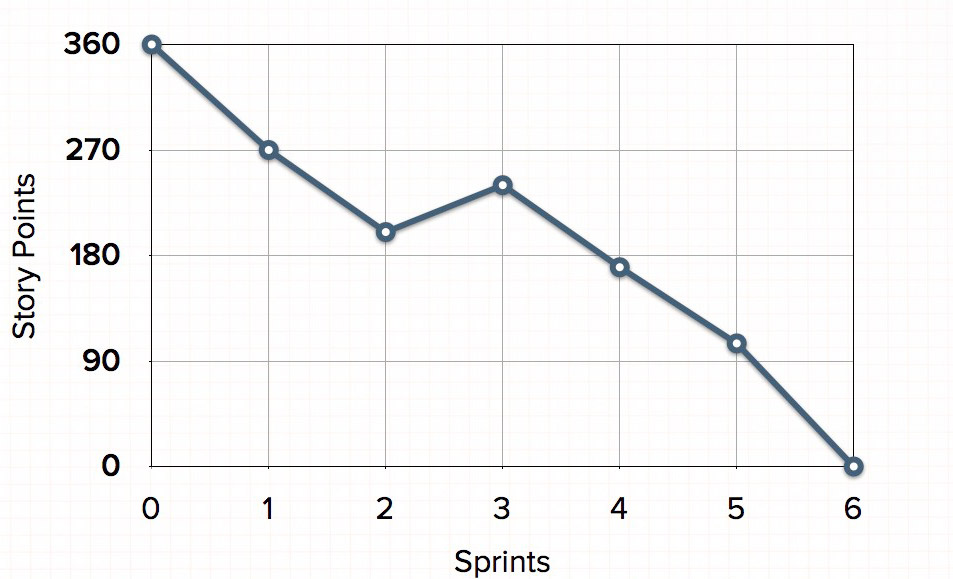
[Make the Product Backlog DEEP](https://www.mountaingoatsoftware.com/blog/make-the-product-backlog-deep)

You can read a great deal more about user stories and agile backlog prioritization in Mike Cohn's,[*User Stories Applied*](https://www.mountaingoatsoftware.com/books/user-stories-applied) book. And check out this [product backlog example](http://www.mountaingoatsoftware.com/scrum/product-backlog-example/).

**Release Burndown Chart**

Progress on a [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) project can be tracked by means of a release burndown chart. The ScrumMaster should update the release burndown chart at the end of each sprint.

The horizontal axis of the sprint burndown chart shows the sprints; the vertical axis shows the amount of work remaining at the start of each sprint. Work remaining can be shown in whatever unit the team prefers -- [story points](https://www.mountaingoatsoftware.com/blog/story-points-are-still-about-effort), ideal days, team days and so on.



On the burndown chart pictured above, the team started a project that was planned to be six sprints. They began with 360 story points of work. To finish within six sprints, they planned to average 60 points per sprint. The first sprint went well and they completed 90, leaving 270.

Things continued to progress well during the second sprint, but during the third sprint, the estimated work remaining actually burned up. This could have been because work was added to the project or because the team changed some estimates of the remaining work. After that, things again went well.

This type of release burndown chart works very well in many situations and for many teams. However, on projects with lots of changing requirements, you may want to look at an alternative release burndown [chart](https://www.mountaingoatsoftware.com/agile/scrum/release-burndown/alternative) as a way of keeping your agile project on track.

The burndown chart is an essential part of any agile project and is a way for the team to clearly see what is happening and how progress is being made during each sprint.

**Sprint Backlog**

The sprint backlog is a list of tasks identified by the Scrum team to be completed during the [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) sprint. During the sprint planning meeting, the team selects some number of product backlog items, usually in the form of user stories, and identifies the tasks necessary to complete each user story. Most teams also estimate how many hours each task will take someone on the team to complete.

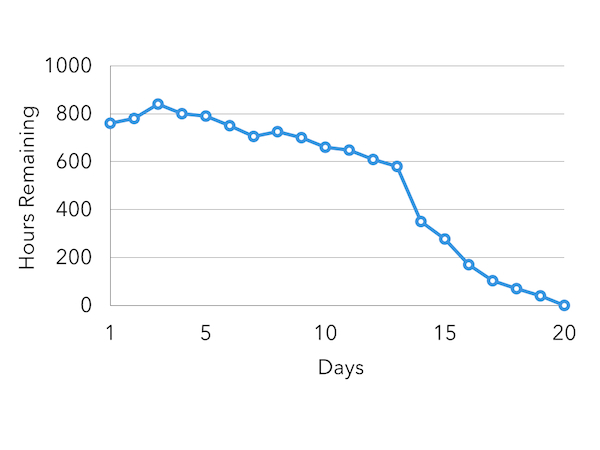
It's critical that the team selects the items and size of the sprint backlog. Because they are the people committing to completing the tasks, they must be the people to choose what they are committing to during the Scrum sprint.

The sprint backlog is commonly maintained as a spreadsheet, but it is also possible to use your defect tracking system or any of a number of software products designed specifically for Scrum or agile. An example of a sprint backlog in a spreadsheet looks like this:



During the Scrum sprint, team members are expected to update the sprint backlog as new information is available, but minimally once per day. Many teams will do this during the daily scrum. Once each day, the estimated work remaining in the sprint is calculated and graphed by the ScrumMaster, resulting in a sprint burndown chart like this one.

The team does its best to pull the right amount of work into the Scrum sprint, but sometimes too much or too little work is pulled in during planning. In this case, the team needs to add or remove tasks.



Let's take an example using the sprint burndown chart above. As you can see, the team in this scenario pulled in too much work initially into the sprint backlog, and still had nearly 600 hours to go on day 13 of a 20-day sprint. The product owner was consulted and agreed to remove some user stories from the sprint. This resulted in the big drop on the chart between days 13 and 14. From there, the team made consistent progress and finished the Scrum sprint successfully.

**Sprint Review Meeting**

In [Scrum](https://www.mountaingoatsoftware.com/agile/scrum), each sprint is required to deliver a potentially shippable product increment. This means that at the end of each sprint, the team has produced a coded, tested and usable piece of software.

So at the end of each sprint, a sprint review meeting is held. During this meeting, the Scrum team shows what they accomplished during the sprint. Typically this takes the form of a demo of the new features.

The sprint review meeting is intentionally kept very informal, typically with rules forbidding the use of PowerPoint slides and allowing no more than two hours of preparation time for the meeting. A sprint review meeting should not become a distraction or significant detour for the team; rather, it should be a natural result of the sprint.

Participants in the sprint review typically include the product owner, the Scrum team, the ScrumMaster, management, customers and developers from other projects.

During the sprint review, the project is assessed against the sprint goal determined during the sprint planning meeting. Ideally, the team has completed each product backlog item brought into the sprint, but it's more important that they achieve the overall goal of the sprint.