The Agile Fundamentals

Agile is a Philosophy guided by the Four Values and Twelve Principle.

We Are Starting With Four Values

Value No. 1 Individuals and Interactions Over Processes and Tools.

- It's Important for People Building a software to interact with Stackeholders(Customer).This not to say that processes and tools are not important they are very important. But being Agile means placing more emphasis on Interactions.
- Interactions between team members or multiple team members are important. Sometimes team members are hung up on how big this software has to be, so what to do, the best way to solve this problem is to have a face-to-face conversation with the team to better develop and have a better estimation of work.

Key takaway here is Interactions are very important between Stackeholders and teammembers.

Value No.2 Working Software Over Comprehensive Documentation.

- This does not mean we need to stop Documentation because without it software navigation becomes very hard. The key word here is Comprehensive.
- What it means to say that focusing more on main deliverable software rather then wasting a lot of time on documentation.
- This can be achieved by **Shared Undestanding**. Having more conversation between team and with the stakeholders and even the customers.
- This shared Undestanding can be achieve again by creating small increaments of software and putting it in the hands of the internal stakeholders or the customer so that they can gain feedback in the software they want.

Key Takeaway here is creating small increatment in software and rasing the shared understanding

of customer and team without too much focusing on documentation.

Value No.3 Customer Collaboration over Contract Negotiation.

- We want to collaborate with customer because we want to delight our customer and also want them to come back to us and rely on our expertise for developing software.
- When we Write Contract we need to keep customer success in mind. So Don't Have strong Contract upfront, collaborate instead to create a win-win situation.
- In Customer Collaration it is actually not fair of development team to ask for all the reqiurement in day one,customer are coming to you in the first place for help, because they want this piece of software to be built by you.

Key Takeaway here Customer Collaboration in software devlopment is important over rigid forced contact and asking requirement for development in small parts over development period.

Value No.4 Responding to Change Over Following a Plan.

- This does not mean have no plan. In Software development we have all kinds of planing. Having a plan is Important.
- Traditionally when we're planing for projects what we are thinking(saying) is that conditions of the universe didnt change between now and the moment we finish it, this rearly happens.
- While developing a complex piece of software somethings will always go wrong or customer requirement change or there may be limitations etc this may render plan useless.

Key Takeaway here Responding to Change in software devlopment is important over just Following the plan. Thing doesnt go as planned always.

Postscript

Agile is really a concept or a philosophy. Off of that, different organizations and companies have built various frameworks that kind of walk you through this step by step of how agile is performed within that various company.

And a lot of those frameworks have become methodology such as **scrum, kanban and scrumban**. And those are the three most popular agile methodologies used by companies today. **In that scrum is the most popular Framework, majority of companies that are utilizing agile are utilizing some version of scrum.**

Kanban is a little bit different way of working through more of a continuous process rather than having various sections and segments that you'll learn about in scrum. Scrumban is really a combination of both. A lot of companies that use scrumban and are using it as a kind of a launching point for them to get from scrum to the kanban methodology.

12 Agile Principles

Agile principles are descriptive guidelines help us make better decisions thought our daily agile activities.

Agile Principle No.1

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

- First thing, is faster or early on; and the second thing, is continuously.
- Try to deliver value faster almost every week or every other week where you can give some kind of valuable software to customer a feature to the customer and they can give you Feedback.

Agile Principle No.2

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

- Responding to change in 21st century is crucial, for any business, even a small business, you need to be able to respond to change.
- So, what this principle is saying is, if somebody is part of an agile team and if they're delivering something to the customer,think about being able to create something for the customer, in case something changes on their business end.
- It's also about delighting customers as well, and really responding to their needs.

Agile Principle No.3

Deliver working software frequently, from a couple weeks to a couple of months,with a preference to the shorter

timescale.

- So, in this principle what we are saying is: focus on the working software. And when we say working software- it is not a requirement; it is not a mockup; it is not a wireframe. **IT IS [A] WORKING SOFTWARE** - something that actually works, a part of the feature that you can give it to the customer, receive feedback, and see how they use it.
- And then the second thing is, the timescale.You are trying to shoot for the shorter timescale a few weeks. And every few weeks, you're trying to have some sort of working software, and you're trying to give that to your customer.
- So, the working software as you can give it to them (customer); so they can get their hands on it. Because most change and most ideas will come out when they actually start working with it and they can see themselves trying to meet that business need. So, that... that's absolutely crucial.

Agile Principle No.4

Business people and developers must work together daily, throughout the project.

- So here what Agile is saying is: There's no need for a creation of this layer. Developers, the team, and the stakeholders or the customers, can actually talk and in fact, it's encouraged, they're talking on almost on a daily basis.
- And the other thing to remember is: we're not talking about business people asking developer when this is going to be done, when this is going to be done? We're talking about a collaboration. What are we building? Has anything changed? So, that kind of communication is happening throughout.
- With agile, it removes the blinders from the developer and allows them to get their questions answered directly from the source rather than going through a layer which doesn't really add much value. Yep, it just helps everybody to get on the same page on what they're building.

Agile Principle No.5

Build projects around motivated individuals. Give them the environment

and support they need, and trust them to get the job done.

- The key concept here is: there's no project manager trying to ask you, you know, what did you do yesterday? or like how much have you done? Like no status reports. The idea here is, you know hire a really good, motivated group of individuals, part of the team who are cross-functional, and actually give them space to actually do amazing work. And when we say work, not just devolve software but also to solve problems for the company or the client.
- So I think the big thing to keep in mind here if you're going to work in an agile team is, regardless of your title these teams are cross-functional. That means, they cross over roles. Your duties and responsibilities are going to evolve and change and that is because, everybody's collaborating together to try to realize that end value.

Agile Principle No.6

The most efficient and effective method of conveying information to and within a development team is face to face conversation.

- What this means is, instead of writing a big piece a requirement document and giving to a developer, we are encouraging more face to face communication. Or even worse creating a ticket, or messaging them, and bothering them all the time, just having meetings face to face, where you are able to communicate what you need.

Agile Principle No.7

Working software is the primary measure of progress.

- This principle is just saying that, whenever you're thinking about progress, let's measure that in terms of our working software. What have we built? Not how much analysis we have done, or how many pages of documentation we have created, or how many mock up pages we've created? This is purely working software.
- And while we may be creating models, as in part of that, and have

them attached to our various items. The working software is really what we're driving towards and what will create the most value.

Agile Principle No.8

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

- In traditional days where business analyst is doing all kind of analysis. They're stressed, and they hit their deadline, they finish, they give it to a developer. Now a development team is stressed out, and they're trying to make the ends meet, hit the deadline, and then finally they're done. Then it goes to Tester. It's Tester's time to be stressful now. They're testing everything, and then implementation time comes in, and everybody stressed out. So, in Agile, what we're trying to do is, instead of this big bang thing wherein everybody's working in phases, we're talking about, like, what are key features? What you need to do?. What is the team's capacity? And think about a sustainable pace that team can develop in.
- I think its really important that because in Agile the whole team is getting together, so, it makes it more sustainable, because it's a cross-functional team, so, everybody is able to help move that that particular project forward to meet the value.

Agile Principle No.9

Continuous attention to technical excellence and good design enhances agility

- This will tackle a little bit about quality in this principle. So, if you build something and if there is no quality in place, you cannot build anything on top of it. So, it's going to be harder for you to be agile in the future. So, agile says: think about quality, think about technical excellence, as you're building certain products.
- You evolve those features and functionality you've built, right? You're making slight changes to them, and additions to them to help drive value later on. If you've got problems early on in that design, or in that initial quality, you're gonna have problems for a

long time.

Agile Principle No.10

Simplicity- the art of maximizing the amount of work not done- is essential.

- When I think about this principle Jeremy, I think about Google's landing page. So when you go to Google's landing page, it is just one search button and it's a white screen. And it is beautiful on the way that it works. Just because you have more and more features, it doesn't make a software pleasant to use. So the idea here, is in Agile, you're also thinking about the outcome, and what ou-... what kind of outcome are you looking. And you can do that by building less and less, and delivering more.
- So, making sure that you're delivering value on everything that you're building, not just building something, or creating something to create it.

Agile Principle No.11

The best architectures, requirements, and designs emerge from self-organizing teams.

- So in agile teams, where the teams are self-organizing and cross-functional, meaning depending on what kind of project you're working on, you're going to have people with the right skill sets. And that's what a team is formed of. Is everybody in the team has unique skill sets and they have what it takes to make this project successful. So, when they need something- a question about an architecture- they already have somebody in that team, so, they can actually talk to somebody, versus borrowing somebody, from somebody else.
- In waterfall, or in other methodologies, a lot of times you need to go ask a manager. "Hey, can I have somebody from your team- from your testing team- to help out with this?". And it usually takes a long time. Where[as] in agile, with it being self-organizing, if you need somebody, you can go tap them on the shoulder and see if they have availability to come help your team, or to join the team potentially.

Agile Principle No.12

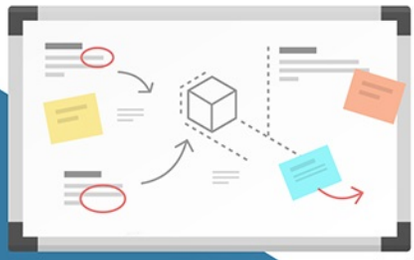
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

- This is actually one of the key principle of agile, that one my favorite principle of agile (instructor)as well. So, as you're working in a product as a team, you're going to learn different things about each other ways of working. So at each interval like every two three weeks, you're getting back and you're looking at what went well, what didn't go well, and how can we improve. and this is a key principle of agile.
- Agile, with it being more iterative, every few weeks, you're actually able to look and see how it's going and make adjustments as necessary, to make the team even more successful.

12 AGILE PRINCIPLES

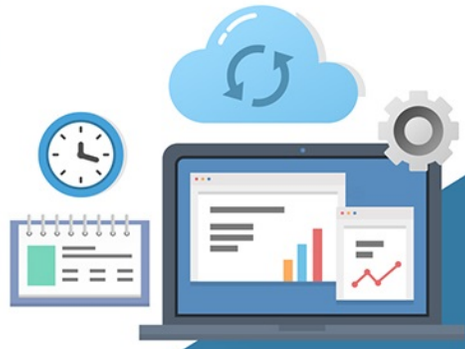
DELIVER VALUE FASTER

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.



WELCOME CHANGE

Welcome changes to requirements, even late in projects. Agile processes harness that change for the customer's competitive advantage.



DELIVER WORKING SOFTWARE FREQUENTLY

Working software should be delivered after a couple of weeks to a couple of months, with a preference to the shorter timescale.

WORK TOGETHER DAILY

Business people and developers must work together daily throughout the project.

BUILD PROJECTS AROUND MOTIVATED INDIVIDUALS

Give them the environment and support they need and trust them to get the job done.

FACE-TO-FACE CONVERSATIONS

The most efficient and effective method of conveying information to and within a development team is with face-to-face conversation.

WORKING SOFTWARE IS KEY

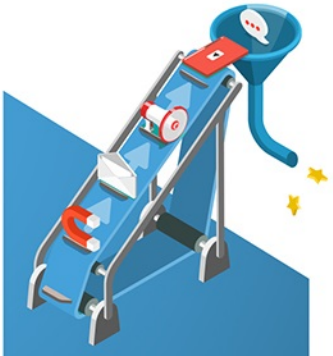
Working software is the primary measure of progress. Not requirements, not models, not designs... working software.

SUSTAINABLE DEVELOPMENT

Agile processes promote sustainable development. The sponsors, project team members (developers), and users should be able to maintain a constant pace indefinitely.

ATTENTION TO TECHNICAL EXCELLENCE

Continuous attention to technical excellence and good design enhances agility.



SIMPLICITY

The art of maximizing the amount of work not done is essential.

SELF-ORGANIZING TEAMS

The best architectures, requirements, and designs emerge from self-organizing teams.

REFLECT AND ADJUST

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

THE AGILE COACH

TheAgileCoach.com

The BAGUIDE
TheBAGuide.com

TheBAGuide.com

Benefits of Agile

Agile was created because of the pitfalls and downfalls of a *traditional waterfall methodology.*You see with waterfall there's various phases and you can't move to the next phase until that previous phase is complete. And once you've moved past a phase, it's very difficult to go back to a previous phase.

So for waterfall you need to identify all the requirements up front, then you need a model and analyze all those requirements. Then you need to develop some type of solution that meets those requirements. Then you need to develop some type of solution that meets those requirements. Then you go through the testing and move to production phases.

But if you're already in the development or testing and you identify some changes, to go back it takes a lot of work for that team to go back to requirements and push that as a separate piece through that waterfall process. As well, because of having to do those things step by step by step the value isn't delivered to the end user and to the customer until the end of the process and so it causes a lot of problem.

As well, **with waterfall it's a much longer period to deliver any type of value or gain any feedback from the users or the customer. that ultimately asked for this solution because you have to**

gather all the requirements, go through design development and testing, that is months, sometimes years to actually receive some type of solution that may or may not meet that business need. Agile was created to help solve those problems.

The number one thing that agile does is it allows you to deliver that **value in smaller increments to the customer**. Not only as a customer that end users get to start utilizing that solution and seeing that value, but also they get to give you feedback. They really like it or they don't really like it or this should be adjusted or the business has changed. And now we said this, but it needs to really be this now.

Agile allows you to do that, you deliver that little bit of value, you gain that critical feedback that will change and adjust the way you move forward on your project. And it really helps to solve a lot of those problems before they become big problems. And because agile is more of an iterative process, the project team is actually able to get feedback on their performance as well. The project team can understand what they could do differently and what they could adjust to make them even more successful and make their solution even more successful.

Some of the challenges to using agile

Well the first thing is agile is actually difficult for existing companies and organizations to implement if they're using some type of methodology like waterfall or other methodology today. And the biggest reason is agile changes everything. It really has to change the whole mindset of the company. It sometimes has to change the organizational structure, has to change the way people and teams in various roles work with each other. **And it really has to be an all or nothing process.**

So a lot of companies will kind of go into and want to move in the agile and they'll do it half heartedly because oh yeah we want to get better and we want our solutions to be out faster and we want to receive all those benefits of using it. But they do it half heartedly and in the end years down the line, they're still using a half agile half not an agile approach. And let's just say it doesn't work.

So that's one of the big challenges is companies have to be all in and ready to spend some money and a lot of time in making that adjustment. Now we're seeing that new companies starting up have a much easier time of making this work. And that's because they don't have existing processes or various things that need to be adjusted. And

that company culture can just be built around that agile philosophy and that agile mindset.

One of the really nice things that a **lot of team members like about agile is the real lack of documentation,** honestly, agile really focuses on more on conversation and communication than writing all of these requirements out in building models and all that stuff. Agile really has that all done through let's sit down, let's discuss it, let's hammer it out and then we'll move forward on whatever is decided there. **That can pose some real challenges when the project's over. And now you have maybe a support team handling this solution that was created. well, there's not any real documentation to tell that support team how it works or some of the common things that come up, it can be a big challenge for them to actually learn and support that particular solution.**

And while we're talking more documentation, **another challenge for the lack of documentation is reusing features or components.** In a traditional methodology such as waterfall you're documenting those features out, your document the design, your document all the analysis points that you've really thought about as you've designed it and as you've developed that solution and that feature, that component can be utilized on additional projects so if you have additional

projects later that are very similar, you can utilize that documentation to implement that feature on that next project. **Because of agile not having much documentation and really that information only being stored or really siloed by the project team members that worked on that particular project, it can be much more difficult in agile to take a feature from a previous project and successfully apply it to the new one.**

And the last challenge we're going to talk about is really there's not a clear role in that agile that takes control or has ownership of that particular project. Instead the team works together collaboratively and everybody chips in and does their part to make sure that the project meets that eventual business need. **The challenge arises when the project goes off course, when it kind of goes into an area that wasn't planned for. Now there's no real role inside of that team to help bring it back. Everybody in the collaborative teams kind of looking at each other and not really sure who should be stepping up to correct the path and get back on plan.**

Scrum

Scrum framework is a very lightweight **framework** that was designed by agilist to **solve complex adaptive problems** that require **sprinting or iterating through solution.**

Scrum's root come from empirical process control framework that values inspection, adaption and transparency. So let's look at inspection and adaption. Inspect and adapt pretty much means that as a team picks up the work and they start working, after some time they can learn more about the problem and the solution and a better way of delivering software.

So scrum says you pick up our problem, you start working on and as you work on it you can look back and inspect and adapt according to what you learn from the process of working itself. Not only you can inspect and adapt on the problems that you're solving, you can inspect and adapt on also how the team is doing and how the team is leveraging the scrum framework.

Transparency on the other hand make scrum framework very effective because all the roles, responsibilities, the meetings and scrum are designed so that team can be more transparent about how they're working and what they're working on.

Let's look at the heart of scrum which is the

sprint. Sprint is just the time selected by the team anywhere between one to four weeks, where they plan the work, they get the work ready, the develop and test and get it production ready so you can put that software in the hands of the customer by the end of the iteration or the sprint.

It's really effective because all the meetings actually happen within the sprint, where do you plan to work, you do a daily checking and you review the work at the end of the timebox or the time that you selected, let's say for example two weeks. And you can also do a retrospective on how the team is doing in terms of working with each other.

Scrum is an agile way of developing software or any project. A lot of times out there people perceive scrum as a methodology, but there's a lot of utility if you look at scrum as a framework. When you look at scrum as a framework, then it's a little bit more than a process. It's more of an adaptive framework that can allow your team to become agile and you can place more value on being agile than just doing perfect scrum. Scrum highly recommend a self-organizing team and cross-functional team. That means if a team is going to take on some work the team members should have the skill sets to be able to develop the solution for their work. If there is a conflict or something that the team doesn't know what to do,

scrum recommends that team discuss among each other and come to a conclusion as a team. **Scrum was initially devolved to manage software products.**

Outside of software development, scrum is used in schools and education, operations, startup, government and marketing. Scrum is unique in his ways because he has very prescriptive roles, responsibilities, ceremonies and artifacts.

Scrum Values

These values are very powerful and effective because scrum is leveraged by a team.And these values really support team members when they are working to solve complex problems.

Openness

Openness just means that team members are open to living the scrum values over just doing scrum. Team members are open to uncover and find effective ways of solving problems. Whenever team members are solved on a tough problems, openness could mean that they are open to inspecting and adapting and figuring out better ways to solve that problem as a team. So being open to other people's ideas and being open to change in general.

Commitment

Whenever people are looking at this value, people

immediately think about commitment to doing the work or executing on the sprint goal or the work that team has committed to, but it's actually more than that. This means that team members are committed to each other in terms of doing their best and coming up with solution collaboratively. This actually also means commitment to not only just delivering software, but delivering software to solve end user's or customer's problems. Commitment also means that team members are going to give their best action and effort to solve a problem versus just focusing on a sprint goal. Focus is another very important scrum value. If you are trying to solve any kind of problems, human beings need to be able to focus. Scrum recommends that team members plan on a certain amount of work and they focus on that for a certain period of time and the whole goal of the team is to get this work that they picked to done.

Focus

Focus is another very important scrum value. If you are trying to solve any kind of problems, human beings need to be able to focus. Scrum recommends that team members plan on a certain amount of work and they focus on that for a certain period of time and the whole goal of the team is to get this work that they picked to done. Focus can also mean that team members are focused on solving a problem that they have

decided as a team to do versus thinking about every shiny thing that's out there and trying to build everything. Focus can also mean that team members are focused on customer happiness and delivering value to the stakeholders

Courage

Mark Twain said: Courage is not the absence of fear, It is acting in spite of it. Courage in this context means that team members show up with courage when they are working on tough problems and are not afraid to try and come up with their bold ideas.Change is usually hard and we are not perfect. However this scrum value encourages team members to do their best despite not being perfect and despite of change being hard.

Respect

Whenever we are working on difficult problems and we are working as a team and we share our success and finish together we will be professional and will respect the team members culture, their background, their ways of working and we will also respect that opinion.

Sprint

Sprint is the heart of the scrum framework. Sprint pretty much just means it is a **time-box, it is a set of time.** Scrum recommends anywhere between **one week to a month where you have** this time-box where **team members** will plan the work along with the **product owner** and **scrum master** and they will **deliver** something called a **product increment** at the **end of the sprint.** So the sprint comprises of all the **scrum ceremonies.** So the **daily standup** happens within the sprint, the **sprint planning** happens within the sprint and the **sprint review and retrospective** it's part of the sprint.

Roles in Scrum framework

Product Owner

One of the main responsibilities of a product owner is to talk to the stakeholders or the users and understand the product vision from them, understand their challenges and what kind of problem they are trying to solve.

Then the product comes back to the team and helps the team understand what are some of the needs and the challenges that the team will be solving with the product that they are developing. And, to do so **product owner writes this very simple description of a feature from an end users**

perspective. And they prioritized that user story in a way where the development team knows exactly which ones to work on.

- So there's a lot of managerial type of stuff that really helps out as a product owner.

Scrum Master

SCRUM MASTER IS NOT A PROJECT MANAGER. HE'S ALSO NOT A TEAM MANAGER. Scrum Master is a **servant-leader role** that is there to **remove the impediments** or the roadblocks in that iteration or time box where the team is very focused on working on those work items. Scrum Master also **coaches teams on the scrum values** to live the scrum values and **facilitate the various scrum meetings or ceremonies.**

- **"What is a standup meeting?" According to The Scrum Guide, "the daily scrum is a 15-minute time-boxed event for the development team" to plan for the next 24 hours.**
- In daily standup the Scrum Master generally asks the following: What did you do yesterday ? What are you going to focus on today ? And do you have impediments(roadblocks) ?.
- They go around and see how everyone is doing. They make a list of impediments so that they can **make visible impediments** or can work with a team member to go out there and

resolve that impediment.
- In an environment where people didn't want to answer those three questions. They do did something called appreciative inquiry. So the question instead of asking 'what did you do yesterday?' what we said is 'what are you going to focus on today? Are we in track to meet our sprint goal?'
- Will facilitate sprint meetings. And do whatever Team needs. Scrum master **facilitate Sprint review.The sprint review is one of the most important ceremonies in Scrum where the team gathers to review completed work and determine whether additional changes are needed.** The official Scrum Guide describes it as a working session and makes the point that the "Scrum team should avoid limiting it to a presentation."

The Development Team (Dev Team)

The Scrum framework says that **everybody else except the product owner and the Scrum Master is a team.** For example, a QA - Quality Assurance Analyst or a business analyst or an architect. All of these roles come under the team. **A team is a professional who can do the work of delivering a potentially releasable increment at the end of the sprint.** The team members are self-organizing,

meaning that they don't need a manager to organize their work. They will actually talk among each other and figure out which I am to work on next. Obviously looking at the priority provided by the product owner.

They **talk to the product owner in almost a daily basis** just to understand the priority and sequencing of the work. And there are also cross-functional. **Cross-functional means they have all the skill sets that are required to develop that product or the feature for that team.**

- In an Agile team, no matter what your title is you're just part of the team and you want to You are a self-organizing team. But we both want to help each other out.
- So when you are a developer, part of the Agile team. You're not saying that you can only do this. You still have to learn other components of what the team was developing.

Scrum ceremonies

Daily stand up or The Daily Scrum meeting

LET ME START WITH WHAT A DAILY STAND UP IS NOT. THIS IS NOT A PROJECT STATUS MEETING WHERE A MANAGER OR A SCRUM MASTER IS THERE TO COLLECT STATUS. This is a quick 15 minutes meeting to do inspect and adapt, which the core of Scrum Framework where everybody comes in and everybody answer the three question. **What did you do yesterday to achieve the sprint goal? What are you going to do today that will help us achieve the sprint goal? And do you have any impediments that might hinder or become an obstacle towards us achieving the sprint goal?** Usually, the team will go one by one and answer these three questions and walk out of them with a very clear plan of attack for the day.

This meeting is held at the same time and same place just to reduce any complexities.This meeting is held at the same time and same place just to reduce any complexities. The Scrum Master facilitates this daily stand up meeting and is responsible for helping the team understand the value of the daily stand up. The team is responsible for having this daily Scrum meeting. The product manager and stakeholders are allowed to come and join the meeting but they are not allowed to interrupt the team. Scrum Master is

responsible for protecting the team from external distraction during the stand up meeting.

In summary, this is the meeting to inspect how the team is doing towards achieving the goal for the sprint and identifying and taking an action item for resolving any impediment or the roadblock.

The reason why this is called a stand-up meeting is that when people are standing up it's harder for them to go on and on because everybody else is also waiting for you to finish.

Sprint planning

So what happens in sprint planning, just like it sounds it is a sprint. **Let's say for this example this is a two week sprint. We are planning this two weeks time-box. We were planning on what goes in used two weeks where we can take in the work and we get this work done done, that this work is shippable to the customer, the customer can use it and give us feedback.**

So when you're planning it, we're not just planning for developing it, we're planning for developing and then testing and kind of making sure this product is ready to be shipped to the customer. Scrum master facilities this meeting meaning he will get the counter invite to everybody and he'll reserve the room and make

sure this time is blocked in everybody's calendar.

Product owner is one of the main players in this meeting. Product owner comes in and he understands the vision of the product. He knows the needs and the challenges of the product. And he's coming back to the team and he's saying that hey I know these items are the **most important items for the stakeholder and this is the priority.** And once the team looks at the priority, they can now say that okay, well based on this priority these are the items that we can, we feel very confident that we can deliver. But this one or two items we're not very sure about or we need to do more research. So then it's **pretty much a back and forth discussion between the team and a product owner.**

Maybe they're negotiating on what should be the scope of the sprint. And after the discussion, let's say it is a two week sprint the planning will be for two hours. **And in these two hours, they create a clear concrete plan on how they're going to work and how they're going to create this potential potentially shippable item in this two weeks iteration.** They will also have already learned about the risk. the dependencies that the work in the sprint might have and they would, they actually had resolved these challenges, risks and impediments for the team to start out to work. The other thing that **they walk out is a very clear**

picture or a goal for the iteration or the sprint.

They know exactly why they are doing this work and it's not necessarily just individual people working on one or two stories, it's for one specific goal that they have. One last thing to remember is a **sprint backlog** comes out of the sprint planning meeting. **Sprint backlog is the amount of work that the team member agreed to work on on that sprint.**

For example, we took a two weeks iteration, let's say they have about twelve user stories that they said if we can get this done in the next two weeks, that is the sprint backlog. And let's just take an example, there are a few scenarios of what happens in the sprint planning meeting. When I was a scrum master, the team had come up with the sprint goal and the team have already committed to a certain amount of user stories and director of a product came back and said oh, we need these two things done right away. So what would you do when you are running scrum and somebody comes back and wants to add more scope to your sprint? One of the things that I did is actually went back to the product owner and I also communicated with the director saying that hey, this is great that this is important. Would you like to talk to our product owner who does the prioritizing of the work? And based on the urgency and the value that it brings to the

organization we can prioritize and put that in the next iteration.

There was another time where this was peak season for this team where they will get a lot of production support or they would have to support the customer base where we knew that, you know, some of the tickets were common. Team members would have to support these users. And one of the things that we did in the sprint planning meeting is we said hey this is already happening. How can we allocate some time, some buffer time so we can also focus on supporting our user base because this is a very peak one month of the year where we need to be there for our customers. So we allocated five story points per sprint for supporting the customer in that sprint.

Sprint planning

The sprint review is held at the last day of the sprint. And this is the event where the team comes together and Scrum Master facilitates this. And invites the Product Owner and the key stakeholders who might be involved in that Sprint work. **Everybody is looking at based on what we did, What are some things that really got done? and what are some things didn't get done? They're also looking at the overall Product Backlog. As the team was working on the Sprint, did there anything else change in the product backlog? that might inform a different priority**

for the next sprint.

So this is a very informal meeting. This is not a project status meeting. This is for the team to showcase on what are some of the progress that they've made. The other thing that's discussed in this meeting is what are potentially uses of this product increment that a team has just developed? What are some other products or competitive companies with similar features are doing? So it's a really good discussion to kind of really look at, what's already done? and how should they evolve the product backlog. And it kind of gives you some key takeaways for the team and the Product Owner. So a Sprint Review is a meeting where you're just revealing what the team did and you want to watch out for some of these things.

If you're working with this new team, chances are at some sprint, the team will say we've got all these things done. We have nothing to demo and we just had got some stories done and we don't have a fully functioning product. Is it important for the team to do Sprint reviews, still? I, as a Scrum Master, would highly recommend teams still do a review even though if it's not fully functional a demo of a product. It's important for stakeholders and the product owner to be there and give some feedback on the work that a team did.

If the team knows that the Sprint review is going

to happen no matter what. The team's focus automatically will be on, Okay, we are already going to have this review. Maybe we should have some kind of increment in the product that we can demo to our stakeholders. And whenever we have time we want to at least have some really good question to ask him. How we are evolving this product? **Another thing that actually happened when I was working as a Scrum Master is a Product Owner or the key stakeholders not being in the Sprint Review where you lose a lot of the value of that meeting when you don't have anybody who can look at the product except the people who built it. And at that point I, as a Scrum Master, had a conversation with the stakeholders and the product owner on being a little bit more available and treating the Sprint Review as a priority.**

Sprint retrospective

Just like it sounds like, it's we're looking at a sprint and we're looking at the retrospective. Retrospective meaning what did the team do well, what didn't go well in this last sprint and what should be improved on. So that's basically what we're trying to capture in this retrospective meeting and this is a two hour meeting for a four weeks sprint. And it could be a one hour meeting for the two week sprint. So scrum master is the one facilitator who sets the stage for the team.

Again creates the safety container where everybody feels empowered to talk about not only the good things, but also some of the bad things or some of the challenges that are happening within the team member or some conflicts that team members have.

Now this next step is gathering data. Like okay, what happened in this last iteration of the sprint? What are some of the challenges that you saw? And everybody will collect data on what happened, you know, what can we learn from this last sprint. And after everybody collects, **these are some of the things that happened in the last sprint, then it's time for the team to generate some insights, like why did this happen, what happened, let's learn more on like the depths of challenges or the problem that the team had. And after they have a good amount of insights, they actually decide on how they want to go about and improve that. So this is an opportunity for the team to have a learning action plan.**

So despite of what happened in the sprint, the team members walk out with a clear plan and what they're going to do and how they're going to improve. **And sometimes the team will vote on what are the three main things that they want to implement or they want to be mindful of in the next sprint.** Retrospective, you're bringing in every two weeks or three weeks, and sometimes

just asking those three question: what went well this sprint? What didn't go well? And what should we improve? Those things can kind of get old and I think it's important for teams to kind of change things up a little bit. I've done things like taking team out on lunch and doing a lunch retro, where it's a little bit light touch, more for conversational retro. And we come up with a few, just a few items on how we want to improve.

One of my favorite ways to facilitate a retro is to do a sailboat retro. So you just draw a sailboat and you draw an anchor and anchor meaning what are some of the impediments. And you draw some wind on the top saying that the winds are some of the things that actually help us and you actually draw a shark in the bottom and the shark would actually imply those are some of the risk that the team might have. And you actually draw out an island asking teams, OK, where is that island? Where do we want to go in the next few months? How do we want to be known in the organization? And what we do and what kind of value we add to the organization?

Another example I'll give you for a retrospective is this is a meeting where we are not there as a team to blame on each other, sometimes things go wrong. If you're working on a complex problem you have to experiment. And whenever you experiment sometimes things go wrong. And one

of the important things as a scrum master, that I did, is I actually created that safe space for everybody to come back and just say what did we really learn, not to blame the people, but what did we learn from what happened and what can we do to improve so that these things don't happen and we make that person recognize the mistake that he made. But it's nothing personal, it's something that we all learn as a team and we move on.

Post Script

Backlog grooming or Backlog refinement

This is one of the things that happened within a sprint, it's called backlog grooming or backlog refinement. So once the sprint is live, you also want to prep for your next sprint. So those stories are already grooming. You're making sure the conversations are happening, the confirmations are happening, the user stories are there so that we could bring it while the sprint is happening, we look at our next dprint and we say OK, these are some of the things that we need to talk about. And you know it's almost like a workshop and you're kind of looking at what kind of problems we need to solve, what are the different potential solutions and you are adding your details to the stories, you know, in the backlog grooming session. Pretty much you're grooming your backlog, you're adding more details to your backlog user stories and getting them ready for the next sprint. Yeah. As a business analyst that's a very interesting point, when you're working as a business analyst or other roles I'm saying business analysts in a very loose term. Yeah. But when you're working in that, you are for

half of that sprint pretty much working in that spirit, and for half of that sprint you're kind of working out of that sprint because you're looking ahead to that, you're going to prep everything, so those user stories are ready, so they're ready to be pulled out. So if you are, most of the time those business analysts, the product owner will get that in there, then it's up to the team to iterate on that and keep checking back with that product owner to make sure that it's in line with their vision when they kind of created that initial user story. So that was always, that was actually always a tough thing as a business analyst is balancing between the current and the next and making sure that you're getting everything done.

SPRINT PLANNING CHECKLIST

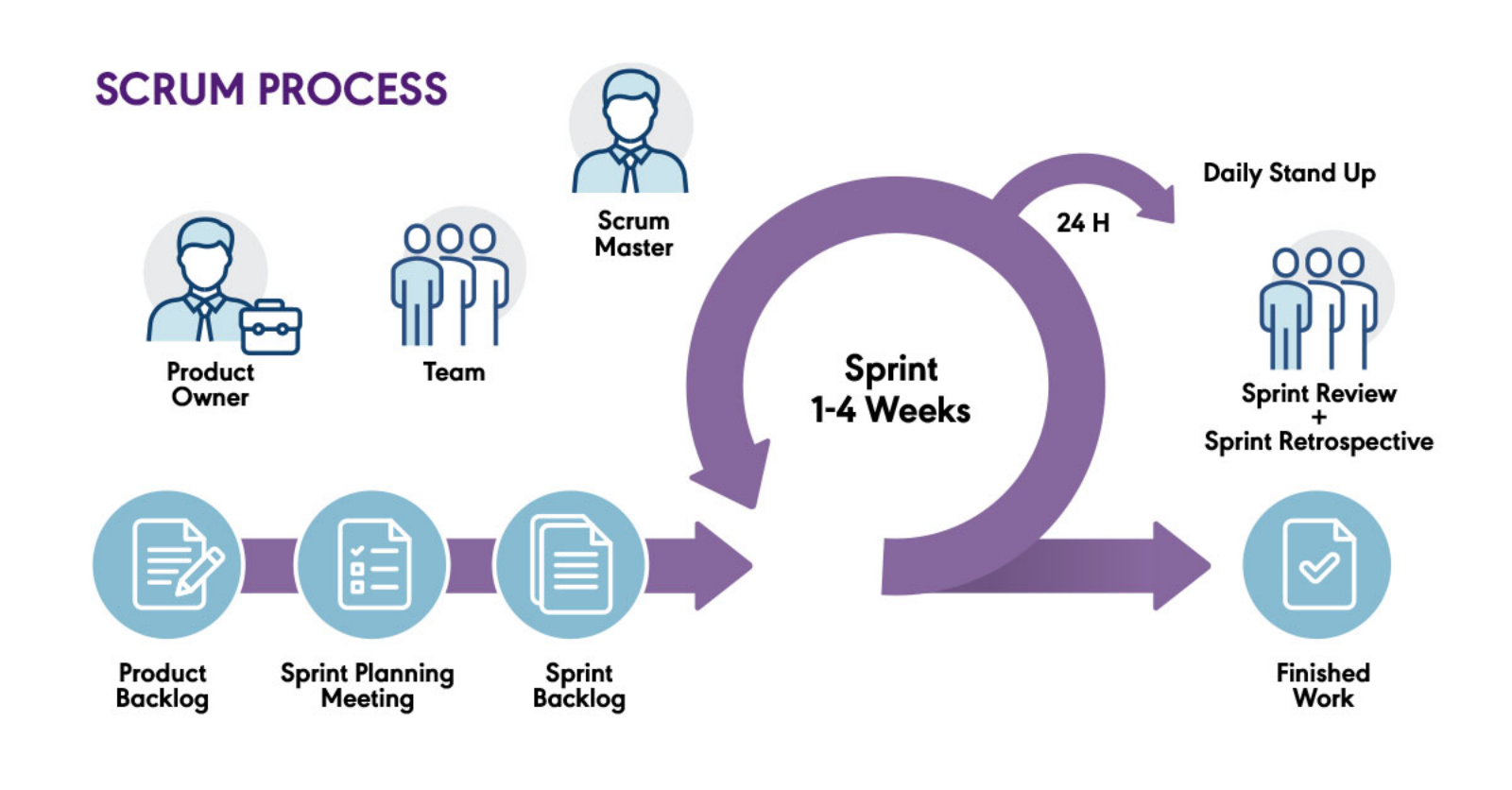
What:

- [ ] Do user stories all contain who, what and why?
- [ ] Are we clear on the acceptance criteria of each story?
- [ ] Have you split large stories into small ones and estimated them?
- [ ] Do you have enough stories for team members to work on for the sprint?
- [ ] Is the priority on the stories clear and understood by everyone?
- [ ] Are the stories on the sprint backlog independent (not dependent upon other stories)?
- [ ] Does the whole team agree on the sprint commitment?
- [ ] Has the team considered PTO (paid time off) and holidays of team members while determining the sprint commitment?
- [ ] Does the team have a Sprint Goal?

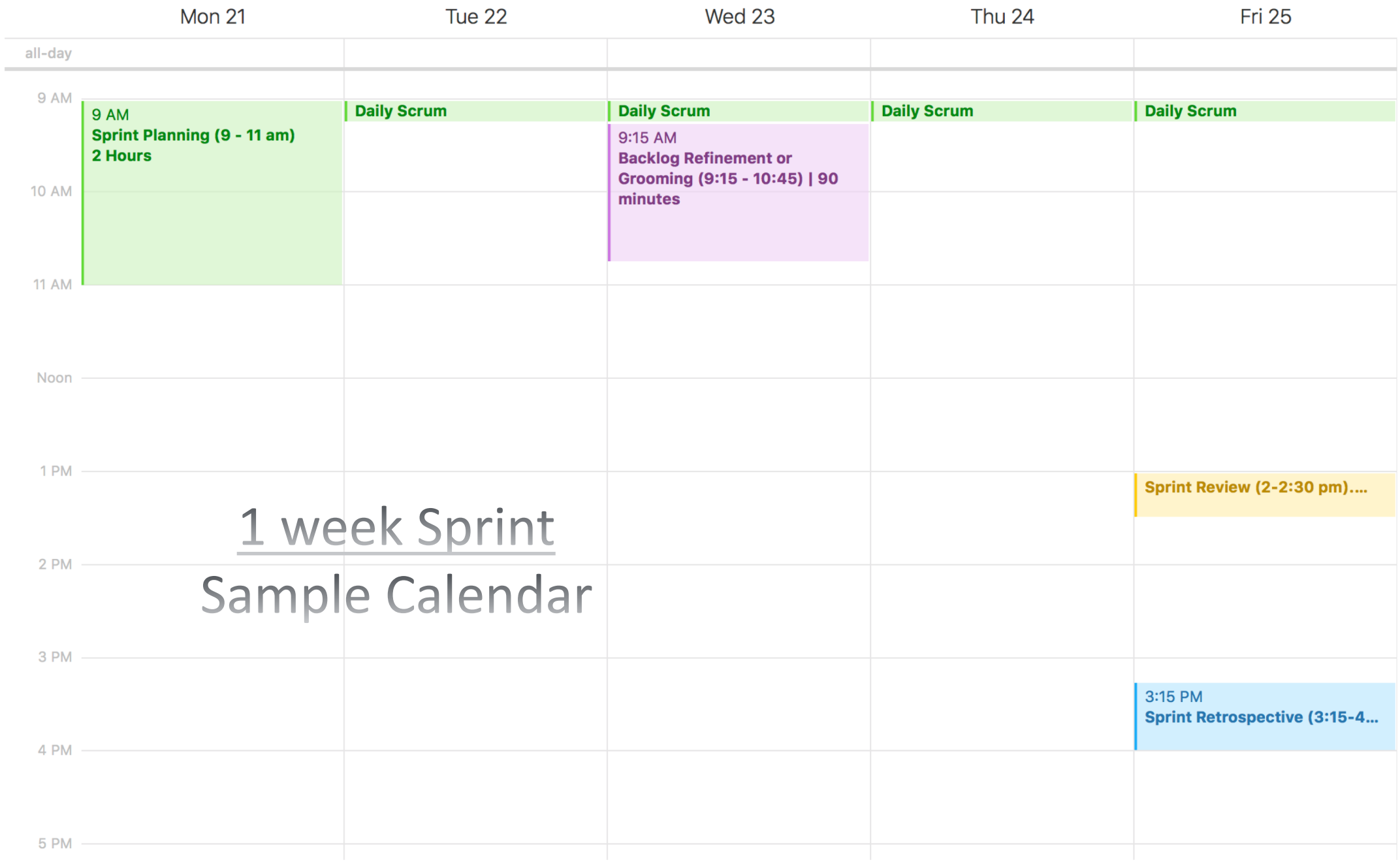
How:

- [ ] Does the team have a shared understanding of Definition of Done?
- [ ] Are there any impediments the team can proactively work on that can help the team to achieve flow?
- [ ] Does every member understand what needs to be done on their respective stories to meet the acceptance criteria?
- [ ] Can team members agree to work on a single story at a time (to avoid context switching)?
- [ ] Are there any knowledge silos or gaps where a task can be done by only one person?
    - [ ] If so, does the team have a plan to share that knowledge with everyone so that everyone in the team can work on that task.
- [ ] Have you put the (most urgent) retrospective action item from the last sprint in this sprint?

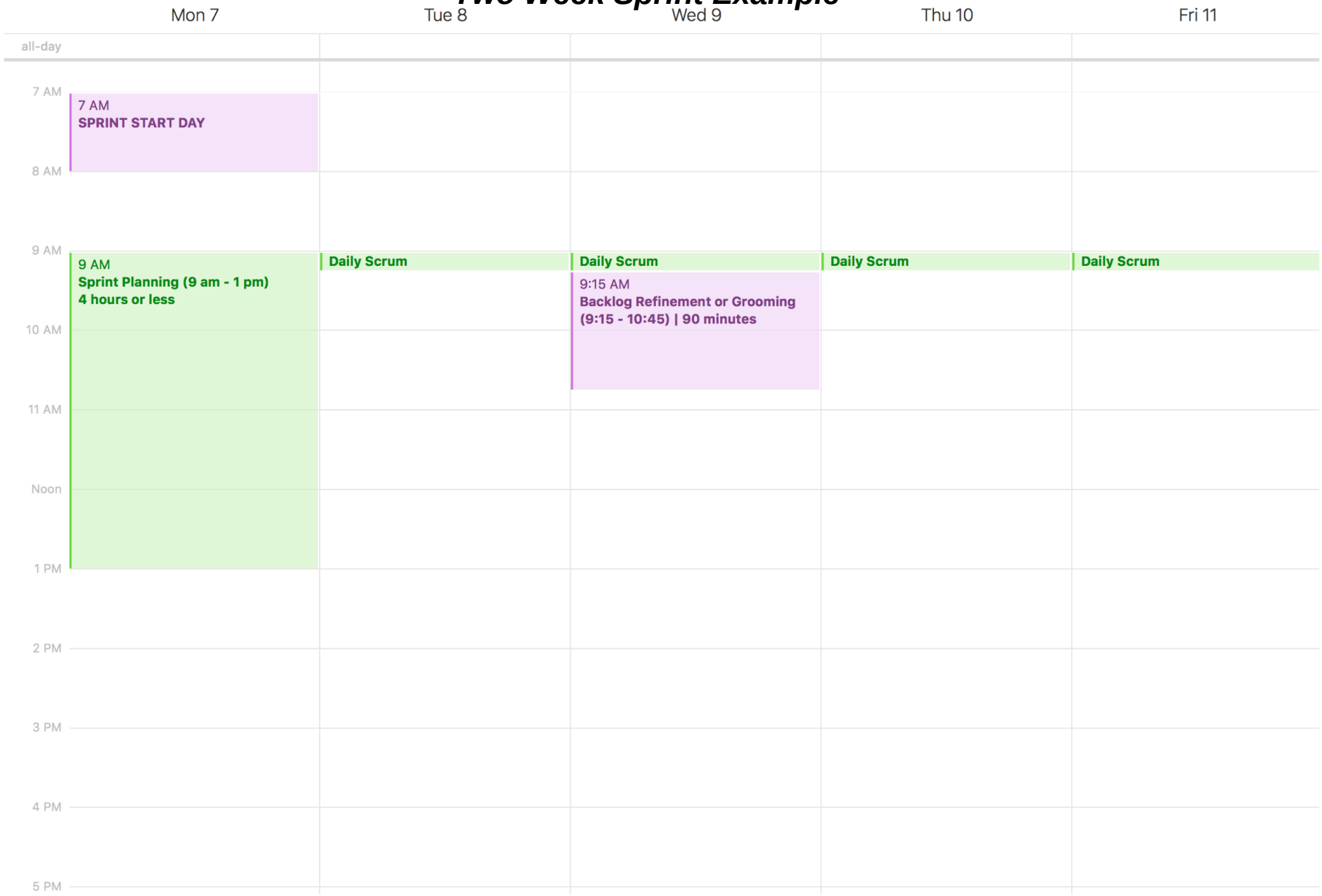
SCRUM PROCESS

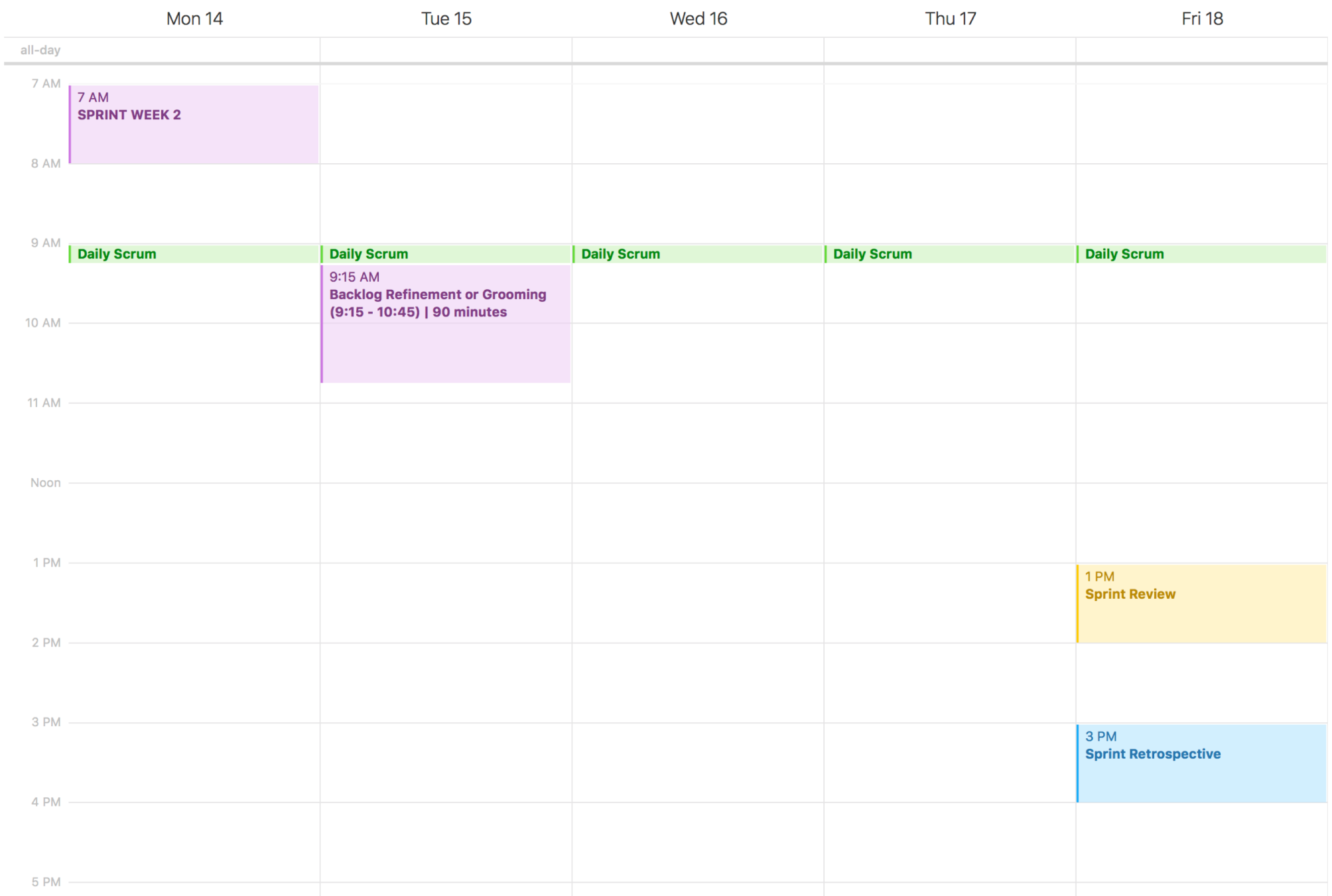


One Week Sprint Example



Two Week Sprint Example




Mon 14
Tue 15
Wed 16
Thu 17
Fri 18
all-day
7 AM
8 AM
9 AM
10 AM
11 AM
Noon
1 PM
2 PM
3 PM
4 PM
5 PM
7 AM
SPRINT WEEK 2
Daily Scrum
Daily Scrum
Daily Scrum
Daily Scrum
Daily Scrum
9:15 AM
Backlog Refinement or Grooming
(9:15 - 10:45) | 90 minutes
1 PM
Sprint Review
3 PM
Sprint Retrospective