Agile - Scrum Methodology

Contains The Manifesto/Principles and the Parts that make up the SCRUM Methodology

# Scrum Roles

## Product owner

* This is the customer/represent the business

## Team Member

* Could be a developer/tester/designer/ba etc (cross functional)

## Scrum Master

* Look at process, look at work process/ Look after the team
* Clean up the task board, remove any obstructions
* Identify any problems and fix the issues, could be pairing
* (“pseudo project manager)

## Stakeholders

* Only involved in product demo
* Feedback goes to the product owner who filters it back to the team

# Scrum Artefacts

## Product Backlog

* Prioritized list of user stories
* Contains the stories that haven’t been done

## Sprint backlog

* Certain amount of stories were sure we can do
* Do they have acceptance criteria?

## Task board

* Minimum has 3 columns. Sprint Backlog/Doing/Done
* Important top/bottom.
* We have a view of what’s going on, who is doing what? Easily visible
* Can add blockers to the board. Visual representation of what’s done.
* WHIP (Work in Progress)

## Burn down chart

* Sprint/Overall/Release Burn down chart

## Burn up chart

* Sprint Burnout chart usually a straight line
* In general, how much works get added on over time

## Definition of done

* It’s a shared agreement between the whole team as to what done means
* \*When you feel like you no longer needs to touch it again\*

# Scrum Meetings and Sprints

The idea is that you will repeat most of these meetings below as the sprint progresses. When one sprint is finished, you start another until the product is signed off as done by the customer, run out of time(shouldn’t happen) or money.

## Sprint Zero / Iteration Zero

Where we box off all the stuff we need to succeed. Set up work. Laying the groundwork so that you can just get on with the development. This could include:

* Set up github/sharepoint.
* Assign roles (Scrum Master etc)
* Make sure everything is setup and ready to use/upload.
* Setting up things that may not have an associated user stories. Programs/Permissions
* Post its, Stationary.
* Database set-up etc

## Scrum Meetings

## Sprint Planning [Stage 1]

* Product owner presents prioritized stories
* Split Large Stories
* Throw out stories not needed
* Discuss acceptance criteria
* Discuss approach
* Add Acceptance Criteria
* Estimate
* User Stories (Look at User Stories)
* Commit to the stories
* (Some could be done in backlog grooming sessions)
* (Getting the stories into the right area)

## Sprint Planning [Stage 2]

* Discuss implementation in details
* Split stories into tasks (process of splitting it down will often bring out other tasks)
* Adjust the commitment if needed
* Agree the sprint

## Daily Stand-up (scrum)

* Daily (every day without fail)
* Only the team are present
* Brief
* Pointed
* Standing up
* Non confrontational
* (Blocker = Obstacle blocking me from doing my work)

## Sprint Review (Demo) (Important)

* Demonstrate the work done
* No powerpoint – working software!
* Anyone can attend
* A chance to show off
* Gather feedback from the stakeholders
* (Product owners gauges reaction of stakeholders/ clarify everything)
* Get signoff on what’s been done (conceptual)
* Provide recommendations to stakeholders

## Sprint Retrospective

* 1 – 2 hours long for a 2 week sprint
* Discuss the process (what problems did we run into, how did the development go?)
* Work out what went well?
* What didn’t go so well?
* What could be improved? (Lessons Learned)

**Remember the Mantra*:*** *“Regardless of what we discover, we understand and truly believe that everyone did the best job they would, given what they knew at the time, their skills and abilities, the resources available and the situation at hand.”*

# Estimation

Commonly used within Stage 1 Planning. Used to estimate how muc resource a given task will take. Uses points and **NOT TIME** to gauge.

## Estimation

Development Spike – Used to research how long it will take to do a task, that’s not been done

before. Research new things to do and estimate how long it will do.

Relative Estimates are better

The bigger something is, the harder it is to estimate

## Story Points

People can use numbers or t-shirt sizes to estimate how long development of a story to do.

Relative Sizing and Arbitrary number

(Commonly use Fibonacci)

## Methods of Estimation

Planning Poker

Team Estimation Game

## Velocity

Number of story points they can do in a given iteration/sprint.

Once we know our velocity, we can make simple judgements for estimation

We can measure it using burn-up/down charts

# Conduct of a Retrospective

THE RETROSPECTIVE PRIME DIRECTIVE | THE RETROSPECTIVE PRIME DIRECTIVE

**“Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available and the situation at hand.”**

THE RETROSPECTIVE PRIME DIRECTIVE | THE RETROSPECTIVE PRIME DIRECTIVE

## The Agenda

### Set the Stage

* Whoever is running it will makes sure were all comfortable and there is no conflicts
* Reminds of the Prime Directive
* And the goal

### Gather Data

* Go back over the week, just get the sprint back in everyone’s head. Get their head in the right place.
* Generating insight, not asking why, just asking what and when

### Generate Insights

* Try to find out all that data means.
* Bad Day/ Bad Code or Bad Code/ Bad Day

### Decide what to do

* What are our actionable points, we want to use it as a bridge
* Some stuff is worth noting, but if you can’t fix it in the next sprint, don’t bother with it
* Identify the things that have the biggest impact, then come up with some action plan

### Close the Retro

* Gonna summarise, outline the action plan, 3 main points explain.
* Appreciation Exercise (newer teams)

### Ways to visually show the Retrospective

* Mood Lines(Cliffs and Valleys)
* The - Retrospective Starfish