Agile, 23/09 (Week 1)

Agile is an umbrella term:

* Embrace change and technical progress
* Iterative (adding on as you go)
* Empirical (measure progress)
* Incremental (progressive change)
* Self-organising teams (left to do your job)
* Empowered (self-motivated)
* Fail fast (the quicker you fail, the quicker you can change your approach)
* Feedback (constant)
* Etc.

Agile manifesto:

* Individuals and interactions over processes and tools (value face to face interaction)
* Working software over comprehensive documentation (some form of product that works over paperwork)
* Customer collaboration over contract negotiation (respond to demands and set realistic targets for your team)
* Responding to change over following a plan (be responsive to the world around you, basically, rather than following a plan to the letter when needed.)

Not that you shouldn’t have a plan but be aware it will change from its original outline.

12 Agile principles:

1. Satisfy customer.
2. Welcome changing requirements.
3. Deliver working software frequently.
4. Business side and developers must work together frequently.
5. Build projects around motivated individuals.
6. Face to face interactions.
7. Working software is the main measure of progress, rather than time deadlines
8. Sustainable development at a constant, achievable rate.
9. Attention to detail.
10. **Simplicity.**
11. Self-organising teams.
12. Regular reflections and evaluations from within the team. (Scrum retrospective)

The Agile framework is designed to guarantee a lower cost, greater quality and a reduced time to market. Compared to the pre-planned features, but the longer timescale and un-determined cost, and less guaranteed quality of the Waterfall model. (The old Crunch model)

The Scrum Framework:

The main framework for our projects:

* Product backlog (things you want to get done)
* Sprint backlog (what you plan to get done over the next 2-4 weeks)
* Potentially Shippable Product Increment (Stage)

Scrum Elements:

Events – The Sprint

* Time based period of software development
* Container for all other events
* One-week to one-month
* Always produce potentially shippable element
* Empirical process control
* Teamwork set of user stories etc

Events – Sprint Planning meeting

* For the Scrum Team to plan Sprint delivery
* 8 hours for one-month Sprint
* Two 4-hour parts, what and how
* Product Owner presents the top priority elements
* All work together pm what functionality is to be developed for the next sprint
* Attendees: Product Owner, Scrum Master, Development Team

Events – Sprint Review

* Informal event held at the end of every Sprint
* Assess progress
* For everyone involved including customers, management and stakeholders
* Inspect what was built at the last sprint and give feedback
* Four-hours per one month sprint
* Attendees: Scrum-Team, Stakeholders invited by Product Owner.

Events – Retrospective

* Held by the Scrum Master to discuss
* What went well, what went wrong, what we could change, actions to be assigned
* Can include discussion on processes, practices, communication, environment, tools etc
* Everyone should get the chance to air their views in an open, honest and constructive way.

Event – Daily Scrum

* Inspect and adapt event for the Development team
* Held by the Scrum Master
* Same time, same place
* 15+ minute time
* A.k.a. The Daily Stand Up
* What did I do yesterday? What did I plan to do today? What got in my way and stopped me from getting done what I wanted to get done?

Scrum artefacts:

* Product backlog: single source of requirements
* Sprint backlog: Sprint to do list
* Increment: Sprint product, ***it must meet the Scrum Team’s definition of done.*** E.g. You’ve done the code, but you’ve copied and pasted it so can you actually explain how it works, so someone else can interact with it and integrate it.
* D: Detailed appropriately
* E: Estimated appropriately
* E: Emergent (liable and responsive to change)
* P: Prioritised

User story: As a <USER>, I want <FEATURE>, so that <EXPLANATION>. For example: As a <CUSTOMER>, I want <TO LOGIN>, so I can <ACCESS SPOTIFY TO LISTEN TO MUSIC>, is a user story.

Scrum Master

* Ensures the Scrum process is followed and understood
* Removes impediments to enable the Devs to be fully functional

Product Owner:

* Owns the Product backlog
* Single authority for the product
* Voice of the customer
* Provides vision and boundaries
* Responsible for profitability and return of investment.

Development team:

* Cross functional, 3 to 9 people.
* Self-empowering
* Self-organised of itself and its work
* Further buzzwords.

Important points:

* Be honest with yourself, others and importantly when things go wrong, so they can be fixed quickly.
* Be open.
* Focus on the priorities set by the PO and SM.
* Critique respectfully.
* If you’re falling behind, say something.
* Remember who you’re working for, and how much you’re being paid to be here.
* TL:DR, ***communicate.***

Working software:

* Helps team gauge its progress, what it’s still got left to complete.
* Can show progress to the PO, stakeholders, and so on.
* Can help you stay ahead of competitors by having something ready to ship

Monitor progress

* Kanban charts, burndown charts.

Velocity

* The amount of work that can realistically be done in a Sprint.
* Based on experience from previous Sprints
* Can be measured in Ideal Days, Story Points
* Junior Devs will get 10-15 Story Points to get done in a week. Senior Devs will get 30+ in a week. For example.
* Helps improve estimates.
* Story points are essentially, a rating system for tasks. Higher values being more complex, lower values being less complex.
* Velocity is not capability, it’s how much the team can get done.
* ***Do. The easy. Tasks. First.*** Hit those targets before attempting the more difficult tasks.

## Points:

* ***Have a Sprint Goal***
* ***Pick items to do most closely linked to the Sprint Goal***
* ***Work out the team’s definition of done***
* ***Have a definition of ready, ready to be worked on, communicate with the PO***
* ***Have a wireframe, a general shape of what the application will look like.***

***Tool reference:***

* Story points use a Fibonacci Sequence to assign values of Story points of cards
* 1, 2, 3, 5, 8, 13, 21, 34
* High ball your estimates. If things go tits up you’re less screwed on your estimates.
* Low = >5, Moderate = 13, High = <21
* Used instead of time as a fairer method of measuring work.

The M(o)S(o)C(o)W method:

* M = Must be Included in the next Sprint
* S = Should be Included in the next Sprint
* C = Could be Included in the next Sprint
* W = Won’t be Included in the next Sprint

CRUD App ideas:

* Something that finds rock music playlists on YT?
* Create a database of rock music playlists
* Add a search function (so for certain bands, or years etc)

By the End of this Week:

* Have your Product Backlog mapped out