Agile Manifesto

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- 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.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Manifesto (Key Points)

- Customer Satisfaction
- Continuous Delivery
- Value
- Welcome Change
- Business + Developer collaboration
- Motivated Individuals
- Trust
- Tools
- Face-to-face communication
- Sustainable Pace
- Technical Excellence
- Design Agility
- Simplicity
- “Work Not Done”
- Self-Organization
- Reflection

Scrum

- Operate in sprints – deliver *something* by sprint end

- Small Teams
- Standups
- Deal with “requirements volatility” and unexpected challenges
- Deliver Quickly
- Iterate
- Respond to emerging requirements and adapt to new technologies and market conditions
- Very “spiral model”-like
- Cross Functional – business plus developers
    - Product Owner
        - Represents the Stakeholders and acts as their proxy
        - “Voice of the Customer”
        - Generate user stories
        - Groom the backlog (but not estimate!)
        - Prioritization
        - Empathetic to all groups and facilitate communication
    - Development Team
        - Not just pure developers – analysts/designers, too
        - Create “Potentially Shippable Increments”
    - Scrum Master
        - Sort of the team lead
        - Minimize distractions for the team
        - Ensure the team follows scrum principles and methodology
        - Often practices servant leadership
- Workflow
    - Sprint Planning
    - Standups/Doing the work
    - Retrospective
    - Backlog refinement
- Backlog
    - Story Points
    - User stories/use cases/items
- Product Increment
    - “Build”
- Burning down
- Definition of Done
- Velocity
- Limitations
    - Time shifted teams
    - Geographically separated teams
    - Skills sets (too focused)
    - Products with External Dependencies
    - Legacy code

Kanban

- Pull in work as necessary
- Limit amount in progress to focus
- Good for legacy/fire-fighting/maintenance work

Extreme Programming (XP)

- Pair Programming
- Emphasis on testability (unit-, acceptance-)
- “Code for today”

Jira (Live Demo)

- Scrum
- Backlog
- Estimation
- Sprint
- Sprint Planning
- Standups
- Burndown
- Retrospective