What is it?

* An adaptive-driven (vs plan-driven / traditional) approach to PM that focuses on adding rapid and regular progress to a well-scoped “ticket”/ work package, with encouragement to evolve planning and delivery of valued features, as they become more understood and concrete, being delivered just-in- time and to scope.

Is it Popular?

* Extremely for S/W and IT projects (or similar) where ambiguity is high.
* Less so for large enterprise projects

Mainstream or Fad?

* Current PNBOK includes Agile elements
* <https://www.pmi.org/learning/library/agile-project-management-pmbok-waterfall-7042>
  + A common misconception in software project management is that in order to properly follow the practices outlined in A Guide to the Project Management Body of Knowledge (PMBOK® Guide)--- (Project Management Institute [PMI]), we must use a non-agile, or waterfall, approach.
  + … fallacy … agile projects still follow the project life cycle and processes as outlined by the PMBOK® Guide.
  + Although the PMBOK® Guide does not dictate methodology, many software project managers nevertheless began to associate the waterfall model with the processes outlined in the PMBOK®Guide.
    - … waterfall was the prevalent methodology at the time
    - … waterfall model provided a framework that supported all of the PMBOK®Guide practices.
* … PMBOK® Guide … states clearly that “there is no single best way to define an ideal project life cycle” (PMI, 2004, p. 20).
  + “the project manager, in collaboration with the project team, is always responsible for determining what processes are appropriate, and the appropriate degree of rigor for each process, for any given project” (PMI, 2004, p. 37).
* PMI does not advocate any particular methodology. It only supplies a standard of good project management practices, and whether individuals choose to follow a waterfall or an agile approach, the PMBOK® Guide will support them both.
* The waterfall methodology outlines the processes of analysis, design, coding, testing, and deployment, which were all done as part of a project. Agile does all of these things within an iteration.

**Fundamentals of Agile - Agile Manifesto:**

**4 Core Values**

1. **Individuals and interactions** over processes and tools
2. **Working s/w** over comprehensive documentation
3. **Customer collaboration** over contract negotiation
4. **Responding to change** rather than following a plan

**12 Principles:**

1. **The highest priority is to satisfy the customer through early and continuous delivery of valuable software.**
   1. Client feedback/corrections come early
   2. Incremental changes occur more quickly and more easily
2. **Working s/w is a good measure of progress**
   1. Incremental functionality delivered, tested, and accepted in pieces rather than 1 big system
   2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
   3. Expect change and manage it
3. **Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. (Sprints, Iterations)**
   1. Continuous improvement – “fail early, fail often”
   2. Short time boxes encourage efficient productivity
4. **Business people and developers must work together daily throughout the project.**
   1. Partnership between developers and business sponsor with both parties feeling responsible for success of deliverables
5. **Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.**
   1. Leadership with strong people-skills vs high pressure deadlines
   2. Empowerment and individual initiative leads to high levels of creativity
6. **The most efficient and effective method of conveying information to and within the development team is face-to-face conversation.**
   1. Face-to-face (as feasible) is one, albeit an important one, form of communication (vs communication primarily via documentation)
7. **Working software is the primary measure of progress.**
   1. Break up project into chunks of functionality where each chunk has a clear definition of what constitutes ‘done’, ‘quality’ attained, user ‘acceptance’.
   2. Do testing more concurrently with s/w development, rather than at end
8. **Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.**
   1. Create an environment where work is sustainable
   2. Success of activity depends on motivation and creativity of team – don’t “sweatshop” them.
9. **Continuous attention to technical excellence and good design enhances agility.**
   1. Doing things the “right way” to avoid unnecessary rework later
   2. “just barely good enough” vs waste, overdesigning or “gold plating” a deliverable.
10. **Simplicity - the art of maximizing the amount of work not done - is essential.**
    1. Minimum viable product – start with something simple and incrementally expand
11. **The best architectures, requirements, and designs emerge from self-organizing teams.**
    1. Collaborative teamwork, by motivated members with complementary skills can deliver superior results over a single person acting alone.
    2. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
12. **Agile is adaptive:**
    1. Design is adaptive to uncertain and changing user requirements
    2. Process can evolve – continuous improvement, short intervals (what’s working / not working), taking corrective actions

Scrum Board:

<https://manifesto.co.uk/agile-concepts-scrum-task-board/>

<https://www.youtube.com/watch?v=Ti2g66b7MUo>

Kanban Board

<https://leankit.com/learn/kanban/kanban-board/>

Little’s Law:

<http://itsadeliverything.com/littles-law-the-basis-of-lean-and-kanban>

<http://www.vissinc.com/2012/09/07/littles-law-isnt-it-a-linear-relationship/>

Scrum vs Kanban:

<https://www.youtube.com/watch?v=rIaz-l1Kf8w>