**Definition of done**

* **User Story Clarity**  
  User stories selected for the sprint are complete with respect to product theme, understood by the team, and have been validated by the detailed acceptance criteria.
* **Tasks Identified**  
  Tasks for selected user stories have been identified and estimated by the team.

These first two topics are typically covered in the sprint planning meeting but are mentioned as done criteria for the sake of verification.

* **Build and package changes**   
  Build and package changes have been communicated to the build master. These changes have been implemented, tested and documented to ensure that they cover the features of the sprint.
* **Product owner approval**  
  Each finished user story has been passed through UAT (User Acceptance Testing) and signed off as meeting requirements
* **Updating Product Backlog**  
  All features not done during the sprint are added back to the product backlog. All incidents/defects not handled during the sprint are added to the product backlog.
* **Environment ready**
* Development environment is ready with all third-party tools configured.
* Staging environment is ready.
* Continuous integration framework is in place. The build engine is configured to schedule hourly, nightly, and release builds.
* Desired build automation is in place. Why "desired"? Because there is no end to build automation :)
* Test data for the selected features has been created
* **Design complete**  
  Design analysis is complete as per the user story or theme. UML diagrams are either created or updated for the feature under development. You might need to prototype various components to ensure that they work together.  Wireframes and prototype have been created and approved by the respective stakeholders.
* **Unit test cases written**  
  Unit test cases have been written for the features to be developed.
* **Documentation Ready**  
  Documentation (Just enough or whatever the team agrees to) to support the sprint demo is ready
* **Pre-Release builds**  
  Pre release builds (hourly/nightly) have been happening and nightly build reports have been published on regular basis. The following could/should be part of pre-release builds:
* Compile and execute unit test cases (mandatory)
* Creation of cross reference of source code
* Execution of automated code reviews for verification of coding rules
* Code coverage reports are generated
* Detection of duplicate source code
* Dependency analysis and generation of design quality matrix (static analysis, cyclomatic complexity)
* Auto deployment in staging environment

It comes down to build automation; there is no end to what can be achieved from automated hourly, nightly builds. The team along with the product owner needs to decide on how much build automation is required.

* **Code Complete**Source code changes are done for all the features in the “to do” list.” Source code has been commented appropriately.
* **Unit testing is done**  
  Unit test cases have been executed and are working successfully
* **Code Refactoring**  
  Source code has been refactored to make it comprehensive, maintainable and, amenable to change. A common mistake is to not keep refactoring in the definition of done. If not taken seriously, refactoring normally spills out to next sprint or, worse, is completely ignored.
* **Code checkin**  
  Source code is checked in the code library with appropriate comments added. If project is using tools which help in maintaining traceability between user stories and the respective source code, proper checkin guidelines are followed.
* **Code merging and tagging**  
  Finalized source code has been merged with the main branch and tagged appropriately (merging and tagging guidelines are to be used)
* **Automated Code reviews**Automated code review has been completed using the supported tools/technologies. Violations have been shared with the team and the team has resolved all discrepancies to adhere to the coding standard. (Automated code reviews should be hooked up with CI builds.)
* **Peer reviews**  
  Peer reviews are done. If pair programming is used, a separate peer review session might not be required.
* **Code coverage is achieved**  
  Code coverage records for each package are available and whatever the team has decided as the minimum benchmark is achieved.
* **Project metrics are ready**  
  Burndown chart has been updated regularly and is up to date.
* **Release Build**
* Build and packaging  
  A Build (successful) is done using continuous integration framework. Change log report has been generated from Code Library and Release notes have been created. Deliverables have been moved to release area.
* Build deployment in staging environment  
  Build deliverables are deployed in staging environment. If it is easy, this step should be automated.
* **Functional testing done**
* Automated testing  
  All types of automated test cases have been executed and a test report has been generated. All incidents/defects are reported.
* Manual testing  
  Quality assurance team has reviewed the reports generated from automation testing and conducted necessary manual test cases to ensure that tests are passing. All incidents/defects are reported.
* Build issues  
  If any integration or build issues are found, the necessary steps are repeated and respective “Done” points are adhered to.
* **Regression testing done**  
  Regression testing is done to ensure that defects have not been introduced in the unchanged area of the software.
* **Performance testing done**A common mistake is to not keep performance testing in the definition of done. This is an important aspect. Most performance issues are design issues and are hard to fix at a later stage.
* **Acceptance testing done**Each finished user story has been passed through UAT (User Acceptance Testing) and signed off as meeting requirements.
* **Closure**  
  All finished user stories/tasks are marked complete/resolved. Remaining hours for task set to zero before closing the task.
* Other necessary/optional activities  
  Anything which is very specific to the project environment  
  These might include:  
  - Security audit sign off  
  - Deployable on multiple platforms such as Windows

**Acceptance Criteria**

**(condition of satisfaction for user story)**

* It serves the purpose of clarifying business requirements / conditions which must be to satify the user for given requirement.
* It applies to specific product backlog item since it clarifies one item.
* Product owner owns acceptance criteria and development team understands them.
* Acceptance criteria are negotiable between product owner and development team.
* Just meeting acceptance criteria may not necessary mean that definition of done is also met.