100 Most Popular Software Testing Terms

Ajoy Kumar Singha
www.ajoysingha.info

Acceptance testing	Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.
Ad hoc testing	Testing carried out informally; no formal test preparation takes place, no recognized test design technique is used, there are no expectations for results and arbitrariness guides the test execution activity.
Agile testing	Testing practice for a project using agile methodologies, such as extreme programming (XP), treating development as the customer of testing and emphasizing the test-first design paradigm.
Alpha testing	Simulated or actual operational testing by potential users/customers or an independent test team at the developers' site, but outside the development organization. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing.
Back-to-back testing	Testing in which two or more variants of a component or system are executed with the same inputs, the outputs compared, and analyzed in cases of discrepancies.
Beta testing	Operational testing by potential and/or existing users/customers at an external site not otherwise involved with the developers, to determine whether or not a component or system satisfies the user/customer needs and fits within the business processes. Beta testing is often employed as a form of external acceptance testing for off-the-shelf software in order to acquire feedback from the market.
Big-bang testing	A type of integration testing in which software elements, hardware elements, or both are combined all at once into a component or an overall system, rather than in stages.
Black-box testing	Testing, either functional or non-functional, without reference to the internal structure of the component or system.
Black-box test design technique	Procedure to derive and/or select test cases based on an analysis of the specification, either functional or non-functional, of a component or system without reference to its internal structure.
Blocked test case	A test case that cannot be executed because the preconditions for its execution are not fulfilled.
Bottom-up testing	An incremental approach to integration testing where the lowest level components are tested first, and then used to facilitate the testing of higher level components. This process is repeated until the component at the top of the hierarchy is tested.
Boundary value	An input value or output value which is on the edge of an equivalence partition or at the smallest incremental distance on either side of an edge, for example the

	minimum or maximum value of a range.
Boundary value analysis	A black box test design technique in which test cases are designed based on boundary values.
Branch testing	A white box test design technique in which test cases are designed to execute branches.
Business process-based testing	An approach to testing in which test cases are designed based on descriptions and/or knowledge of business processes.
Capture/playback tool	A type of test execution tool where inputs are recorded during manual testing in order to generate automated test scripts that can be executed later (i.e. replayed). These tools are often used to support automated regression testing.
Certification	The process of confirming that a component, system or person complies with its specified requirements, e.g. by passing an exam.
Code coverage	An analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g. statement coverage, decision coverage or condition coverage.
Compliance testing	The process of testing to determine the compliance of the component or system.
Component integration testing	Testing performed to expose defects in the interfaces and interaction between integrated components.
Condition testing	A white box test design technique in which test cases are designed to execute condition outcomes.
Conversion testing	Testing of software used to convert data from existing systems for use in replacement systems.
Data driven testing	A scripting technique that stores test input and expected results in a table or spreadsheet, so that a single control script can execute all of the tests in the table. Data driven testing is often used to support the application of test execution tools such as capture/playback tools.
Database integrity testing	Testing the methods and processes used to access and manage the data(base), to ensure access methods, processes and data rules function as expected and that during access to the database, data is not corrupted or unexpectedly deleted, updated or created.
Defect	A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.
Defect masking	An occurrence in which one defect prevents the detection of another.

Defect report	A document reporting on any flaw in a component or system that can cause the component or system to fail to perform its required function.
Development testing	Formal or informal testing conducted during the implementation of a component or system, usually in the development environment by developers.
Driver	A software component or test tool that replaces a component that takes care of the control and/or the calling of a component or system.
Equivalence partitioning	A black box test design technique in which test cases are designed to execute representatives from equivalence partitions. In principle test cases are designed to cover each partition at least once.
Error	A human action that produces an incorrect result.
Error guessing	A test design technique where the experience of the tester is used to anticipate what defects might be present in the component or system under test as a result of errors made, and to design tests specifically to expose them.
Exhaustive testing	A test approach in which the test suite comprises all combinations of input values and preconditions.
Exploratory testing	An informal test design technique where the tester actively controls the design of the tests as those tests are performed and uses information gained while testing to design new and better tests.
Failure	Deviation of the component or system from its expected delivery, service or result.
Functional test design technique	Procedure to derive and/or select test cases based on an analysis of the specification of the functionality of a component or system without reference to its internal structure.
Functional testing	Testing based on an analysis of the specification of the functionality of a component or system.
Functionality testing	The process of testing to determine the functionality of a software product.
Heuristic evaluation	A static usability test technique to determine the compliance of a user interface with recognized usability principles (the so-called "heuristics").
High level test case	A test case without concrete (implementation level) values for input data and expected results. Logical operators are used; instances of the actual values are not yet defined and/or available.
ISTQB	International Software Testing Qualification Board. Click here for more details.
Incident management tool	A tool that facilitates the recording and status tracking of incidents. They often have workflow-oriented facilities to track and control the allocation, correction and re-testing of incidents and provide reporting facilities.

Installability testing	The process of testing the installability of a software product.
Integration testing	Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.
Isolation testing	Testing of individual components in isolation from surrounding components, with surrounding components being simulated by stubs and drivers, if needed.
Keyword driven testing	A scripting technique that uses data files to contain not only test data and expected results, but also keywords related to the application being tested. The keywords are interpreted by special supporting scripts that are called by the control script for the test.
Load testing	A test type concerned with measuring the behavior of a component or system with increasing load, e.g. number of parallel users and/or numbers of transactions to determine what load can be handled by the component or system.
Low level test case	A test case with concrete (implementation level) values for input data and expected results. Logical operators from high level test cases are replaced by actual values that correspond to the objectives of the logical operators.
Maintenance testing	Testing the changes to an operational system or the impact of a changed environment to an operational system.
Monkey testing	Testing by means of a random selection from a large range of inputs and by randomly pushing buttons, ignorant on how the product is being used.
Negative testing	Tests aimed at showing that a component or system does not work. Negative testing is related to the testers' attitude rather than a specific test approach or test design technique, e.g. testing with invalid input values or exceptions.
Non-functional testing	Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, maintainability and portability.
Operational testing	Testing conducted to evaluate a component or system in its operational environment.
Pair testing	Two persons, e.g. two testers, a developer and a tester, or an end-user and a tester, working together to find defects. Typically, they share one computer and trade control of it while testing.
Peer review	A review of a software work product by colleagues of the producer of the product for the purpose of identifying defects and improvements. Examples are inspection, technical review and walkthrough.
Performance testing	The process of testing to determine the performance of a software product.
Portability testing	The process of testing to determine the portability of a software product.
Post-execution	Comparison of actual and expected results, performed after the software has

comparison	finished running.
Priority	The level of (business) importance assigned to an item, e.g. defect.
Quality assurance	Part of quality management focused on providing confidence that quality requirements will be fulfilled.
Random testing	A black box test design technique where test cases are selected, possibly using a pseudo-random generation algorithm, to match an operational profile. This technique can be used for testing non-functional attributes such as reliability and performance.
Recoverability testing	The process of testing to determine the recoverability of a software product.
Regression testing	Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.
Requirements-based testing	An approach to testing in which test cases are designed based on test objectives and test conditions derived from requirements, e.g. tests that exercise specific functions or probe non-functional attributes such as reliability or usability.
Re-testing	Testing that runs test cases that failed the last time they were run, in order to verify the success of corrective actions.
Risk-based testing	An approach to testing to reduce the level of product risks and inform stakeholders on their status, starting in the initial stages of a project. It involves the identification of product risks and their use in guiding the test process.
Severity	The degree of impact that a defect has on the development or operation of a component or system.
Site acceptance testing	Acceptance testing by users/customers at their site, to determine whether or not a component or system satisfies the user/customer needs and fits within the business processes, normally including hardware as well as software.
Smoke test	A subset of all defined/planned test cases that cover the main functionality of a component or system, to ascertaining that the most crucial functions of a program work, but not bothering with finer details. A daily build and smoke test is among industry best practices.
Statistical testing	A test design technique in which a model of the statistical distribution of the input is used to construct representative test cases.
Stress testing	Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements.
Stub	A skeletal or special-purpose implementation of a software component, used to

	develop or test a component that calls or is otherwise dependent on it. It replaces a called component.
Syntax testing	A black box test design technique in which test cases are designed based upon the definition of the input domain and/or output domain.
System integration testing	Testing the integration of systems and packages; testing interfaces to external organizations (e.g. Electronic Data Interchange, Internet).
System testing	The process of testing an integrated system to verify that it meets specified requirements.
Test automation	The use of software to perform or support test activities, e.g. test management, test design, test execution and results checking.
Test case specification	A document specifying a set of test cases (objective, inputs, test actions, expected results, and execution preconditions) for a test item.
Test design specification	A document specifying the test conditions (coverage items) for a test item, the detailed test approach and identifying the associated high level test cases.
Test environment	An environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.
Test harness	A test environment comprised of stubs and drivers needed to execute a test.
Test log	A chronological record of relevant details about the execution of tests.
Test management tool	A tool that provides support to the test management and control part of a test process. It often has several capabilities, such as testware management, scheduling of tests, and the logging of results, progress tracking, incident management and test reporting.
Test oracle	A source to determine expected results to compare with the actual result of the software under test. An oracle may be the existing system (for a benchmark), a user manual, or an individual's specialized knowledge, but should not be the code.
Test plan	A document describing the scope, approach, resources and schedule of intended test activities. It identifies amongst others test items, the features to be tested, the testing tasks, who will do each task, degree of tester independence, the test environment, the test design techniques and entry and exit criteria to be used, and the rationale for their choice, and any risks requiring contingency planning. It is a record of the test planning process.
Test strategy	A high-level description of the test levels to be performed and the testing within those levels for an organization or programme (one or more projects).
Test suite	A set of several test cases for a component or system under test, where the post condition of one test is often used as the precondition for the next one.

Testware	Artifacts produced during the test process required to plan, design, and execute tests, such as documentation, scripts, inputs, expected results, set-up and clear-up procedures, files, databases, environment, and any additional software or utilities used in testing.
Thread testing	A version of component integration testing where the progressive integration of components follows the implementation of subsets of the requirements, as opposed to the integration of components by levels of a hierarchy.
Top-down testing	An incremental approach to integration testing where the component at the top of the component hierarchy is tested first, with lower level components being simulated by stubs. Tested components are then used to test lower level components. The process is repeated until the lowest level components have been tested.
Traceability	The ability to identify related items in documentation and software, such as requirements with associated tests. See also horizontal traceability, vertical traceability.
Usability testing	Testing to determine the extent to which the software product is understood, easy to learn, easy to operate and attractive to the users under specified conditions.
Use case	A sequence of transactions in a dialogue between a user and the system with a tangible result.
Use case testing	A black box test design technique in which test cases are designed to execute user scenarios.
Unit test framework	A tool that provides an environment for unit or component testing in which a component can be tested in isolation or with suitable stubs and drivers. It also provides other support for the developer, such as debugging capabilities.
Validation	Confirmation by examination and through provision of objective evidence that the requirements for a specific intended use or application have been fulfilled.
Verification	Confirmation by examination and through provision of objective evidence that specified requirements have been fulfilled.
Vertical traceability	The tracing of requirements through the layers of development documentation to components.
Volume testing	Testing where the system is subjected to large volumes of data.
Walkthrough	A step-by-step presentation by the author of a document in order to gather information and to establish a common understanding of its content.

White-box testing	Testing based on an analysis of the internal structure of the component or system.

For a complete list of glossary please refer to http://ajoysingha.info/glossary.aspx