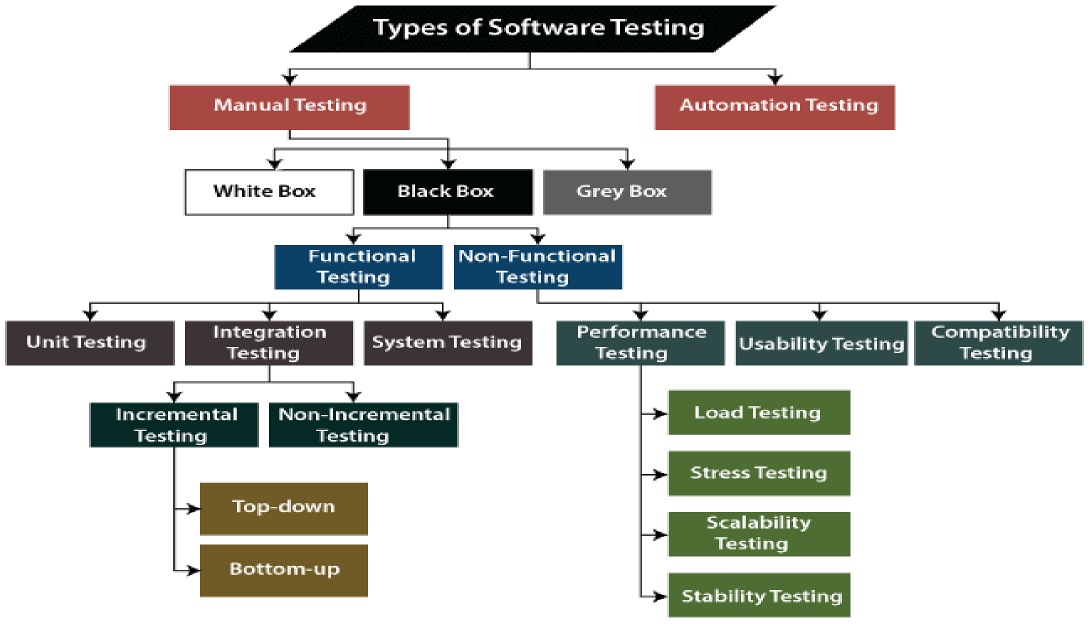
Types of Software Testing

In this section, we are going to understand the various types of software testing, which can be used at the time of the Software Development Life Cycle.

As we know, **software testing** is a process of analyzing an application's functionality as per the customer prerequisite.

If we want to ensure that our software is bug-free or stable, we must perform the various types of software testing because testing is the only method that makes our application bug-free.

.



The different types of Software Testing

The categorization of software testing is a part of diverse testing activities, such as **test strategy, test deliverables, a defined test objective, etc**. And software testing is the execution of the software to find defects.

The purpose of having a testing type is to confirm the **AUT** (Application Under Test).

To start testing, we should have a **requirement, application-ready, necessary resources available**. To maintain accountability, we should assign a respective module to different test engineers.

Manual Testing

Manual testing is a software testing process in which test cases are executed manually without using any automated tool. All test cases executed by the tester manually according to the end user's perspective. It ensures whether the application is working, as mentioned in the requirement document or not. Test cases are planned and implemented to complete almost 100 percent of the software application. Test case reports are also generated manually.

Manual Testing is one of the most fundamental testing processes as it can find both visible and hidden defects of the software. The difference between expected output and output, given by the software, is defined as a defect. The developer fixed the defects and handed it to the tester for retesting.

Manual testing is mandatory for every newly developed software before automated testing. This testing requires great efforts and time, but it gives the surety of bug-free software. Manual Testing requires knowledge of manual testing techniques but not of any automated testing tool.

Manual testing is essential because one of the software testing fundamentals is "100% automation is not possible."

Why we need manual testing

Whenever an application comes into the market, and it is unstable or having a bug or issues or creating a problem while end-users are using it.

If we don't want to face these kinds of problems, we need to perform one round of testing to make the application bug free and stable and deliver a quality product to the client, because if the application is bug free, the end-user will use the application more conveniently.

If the test engineer does manual testing, he/she can test the application as an end-user perspective and get more familiar with the product, which helps them to write the correct test cases of the application and give the quick feedback of the application.

Types of Manual Testing

There are various methods used for manual testing. Each technique is used according to its testing criteria. Types of manual testing are given below:

- White Box Testing
- Black Box Testing
- Gray Box Testing

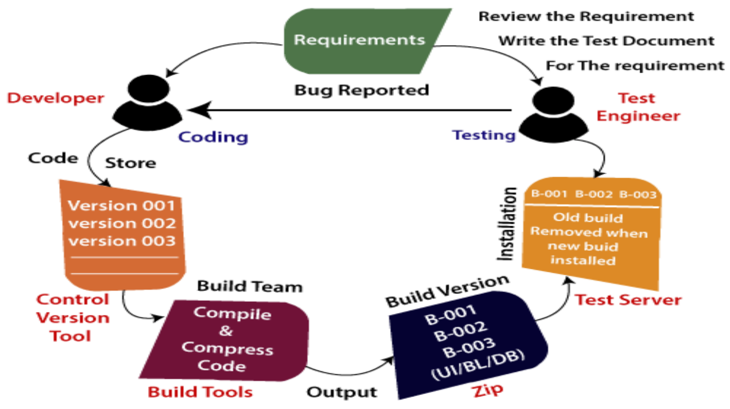
How to perform Manual Testing

- First, tester observes all documents related to software, to select testing areas.
- Tester analyses requirement documents to cover all requirements stated by the customer.
- Tester develops the test cases according to the requirement document.
- All test cases are executed manually by using Black box testing and white box testing.

- If bugs occurred then the testing team informs the development team.
- The Development team fixes bugs and handed software to the testing team for a retest.

Software Build Process

- Once the requirement is collected, it will provide to the two different team development and testing team.
- After getting the requirement, the concerned developer will start writing the code.
- And in the meantime, the test engineer understands the requirement and prepares the required documents, up to now the developer may complete the code and store in the **Control Version tool**.
- After that, the code changes in the UI, and these changes handle by one separate team, which is known as the **build team**.
- This build team will take the code and start compile and compress the code with the help of a build tool. Once we got some output, the output goes in the zip file, which is known as **Build** (application or software).Each Build will have some unique number like (B001, B002).
- Then this particular Build will be installed in the test server. After that, the test engineer will access this test server with the help of the Test URL and start testing the application.
- If the test engineer found any bug, he/she will be reported to the concerned developer.
- Then the developer will reproduce the bug in the test server and fix the bug and again store the code in the Control version tool, and it will install the new updated file and remove the old file; this process is continued until we get the stable Build.
- Once we got the stable Build, it will be handed over to the customer.



Software Build Process

Note1

- Once we collect the file from the Control version tool, we will use the build tool to compile the code from high-level language to machine level language. After compilation, if the file size will increase, so we will compress that particular file and dumped into the test server.

- This process is done by **Build team**, **developer** (if build team is not there, a developer can do it) or the **test lead** (if the build team directly handle the zip and install the application to the test server and inform the test engineer).
- Generally, we can't get a new Build for every bug; else, most of the time will be wasted only in creating the builds.

Note2

Build team

The main job of the build team is to create the application or the Build and converting the high-level language into low-level language.

Build

It is software, which is used to convert the code into application format. And it consists of some set of features and bug fixes that are handed over to the test engineer for testing purposes until it becomes stable.

Control version tool

It is a software or application, which is used for the following purpose:

- In this tool, we can save different types of files.
- It is always secured because we access the file from the tools using the same login credentials.
- The primary objective of the tools is to track the changes done for the existing files.

Advantages of Manual Testing

- It does not require programming knowledge while using the Black box method.
- It is used to test dynamically changing GUI designs.
- Tester interacts with software as a real user so that they are able to discover usability and user interface issues.
- It ensures that the software is a hundred percent bug-free.
- It is cost-effective.
- Easy to learn for new testers.

Disadvantages of Manual Testing

- It requires a large number of human resources.
- It is very time-consuming.
- Tester develops test cases based on their skills and experience. There is no evidence that they have covered all functions or not.

- Test cases cannot be used again. Need to develop separate test cases for each new software.
- It does not provide testing on all aspects of testing.
- Since two teams work together, sometimes it is difficult to understand each other's motives, it can mislead the process.

White Box Testing

The box testing approach of software testing consists of black box testing and white box testing. We are discussing here white box testing which also known as glass box is **testing, structural testing, clear box testing, open box testing and transparent box testing**. It tests internal coding and infrastructure of a software focus on checking of predefined inputs against expected and desired outputs. It is based on inner workings of an application and revolves around internal structure testing. In this type of testing programming skills are required to design test cases. The primary goal of white box testing is to focus on the flow of inputs and outputs through the software and strengthening the security of the software.

The term 'white box' is used because of the internal perspective of the system. The clear box or white box or transparent box name denote the ability to see through the software's outer shell into its inner workings.

Developers do white box testing. In this, the developer will test every line of the code of the program. The developers perform the White-box testing and then send the application or the software to the testing team, where they will perform the black box testing and verify the application along with the requirements and identify the bugs and sends it to the developer.

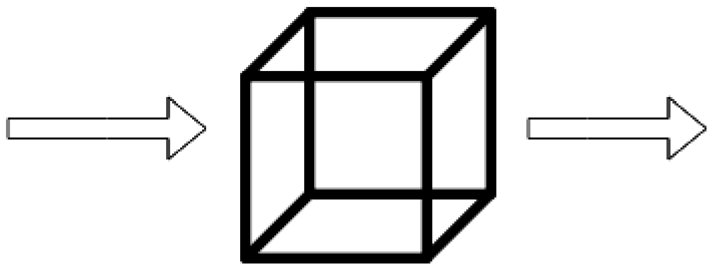
The developer fixes the bugs and does one round of white box testing and sends it to the testing team. Here, fixing the bugs implies that the bug is deleted, and the particular feature is working fine on the application.

Here, the test engineers will not include in fixing the defects for the following reasons:

- Fixing the bug might interrupt the other features. Therefore, the test engineer should always find the bugs, and developers should still be doing the bug fixes.
- If the test engineers spend most of the time fixing the defects, then they may be unable to find the other bugs in the application.

The white box testing contains various tests, which are as follows:

- Path testing
- Loop testing
- Condition testing
- Testing based on the memory perspective
- Test performance of the program



Whitebox Testing

White box testing follows some working steps to make testing manageable and easy to understand what the next task to do. There are some basic steps to perform white box testing.

Generic steps of white box testing

- Design all test scenarios, test cases and prioritize them according to high priority number.
- This step involves the study of code at runtime to examine the resource utilization, not accessed areas of the code, time taken by various methods and operations and so on.
- In this step testing of internal subroutines takes place. Internal subroutines such as nonpublic methods, interfaces are able to handle all types of data appropriately or not.
- This step focuses on testing of control statements like loops and conditional statements to check the efficiency and accuracy for different data inputs.
- In the last step white box testing includes security testing to check all possible security loopholes by looking at how the code handles security.

Reasons for white box testing

- It identifies internal security holes.
- To check the way of input inside the code.
- Check the functionality of conditional loops.
- To test function, object, and statement at an individual level.

Advantages of White box testing

- White box testing optimizes code so hidden errors can be identified.
- Test cases of white box testing can be easily automated.
- This testing is more thorough than other testing approaches as it covers all code paths.
- It can be started in the SDLC phase even without GUI.

Disadvantages of White box testing

- White box testing is too much time consuming when it comes to large-scale programming applications.

- White box testing is much expensive and complex.
- It can lead to production error because it is not detailed by the developers.
- White box testing needs professional programmers who have a detailed knowledge and understanding of programming language and implementation.

Techniques Used in White Box Testing

Data Flow Testing	Data flow testing is a group of testing strategies that examines the control flow of programs in order to explore the sequence of variables according to the sequence of events.
Control Flow Testing	Control flow testing determines the execution order of statements or instructions of the program through a control structure. The control structure of a program is used to develop a test case for the program. In this technique, a particular part of a large program is selected by the tester to set the testing path. Test cases represented by the control graph of the program.
Branch Testing	Branch coverage technique is used to cover all branches of the control flow graph. It covers all the possible outcomes (true and false) of each condition of decision point at least once.
Statement Testing	Statement coverage technique is used to design white box test cases. This technique involves execution of all statements of the source code at least once. It is used to calculate the total number of executed statements in the source code, out of total statements present in the source code.
Decision Testing	This technique reports true and false outcomes of Boolean expressions. Whenever there is a possibility of two or more outcomes from the statements like do while statement, if statement and case statement (Control flow statements), it is considered as decision point because there are two outcomes either true or false.

Difference between white-box testing and black-box testing

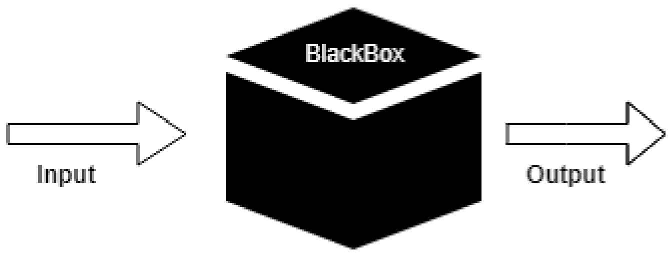
Following are the significant differences between white box testing and black box testing:

White-box testing	Black box testing
The developers can perform white box testing.	The test engineers perform the black box testing.
To perform WBT, we should have an understanding of the programming languages.	To perform BBT, there is no need to have an understanding of the programming languages.
In this, we will look into the source code and test the logic of the code.	In this, we will verify the functionality of the application based on the requirement specification.
In this, the developer should know about the internal design of the code.	In this, there is no need to know about the internal design of the code.

Black box testing

Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.

In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not. If the function produces correct output, then it is passed in testing, otherwise failed. The test team reports the result to the development team and then tests the next function. After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.



Generic steps of black box testing

- The black box test is based on the specification of requirements, so it is examined in the beginning.

- In the second step, the tester creates a positive test scenario and an adverse test scenario by selecting valid and invalid input values to check that the software is processing them correctly or incorrectly.
- In the third step, the tester develops various test cases such as decision table, all pairs test, equivalent division, error estimation, cause-effect graph, etc.
- The fourth phase includes the execution of all test cases.
- In the fifth step, the tester compares the expected output against the actual output.
- In the sixth and final step, if there is any flaw in the software, then it is cured and tested again.

Test procedure

The test procedure of black box testing is a kind of process in which the tester has specific knowledge about the software's work, and it develops test cases to check the accuracy of the software's functionality.

It does not require programming knowledge of the software. All test cases are designed by considering the input and output of a particular function.A tester knows about the definite output of a particular input, but not about how the result is arising. There are various techniques used in black box testing for testing like decision table technique, boundary value analysis technique, state transition, All-pair testing, cause-effect graph technique, equivalence partitioning technique, error guessing technique, use case technique and user story technique. All these techniques have been explained in detail within the tutorial.

Test cases

Test cases are created considering the specification of the requirements. These test cases are generally created from working descriptions of the software including requirements, design parameters, and other specifications. For the testing, the test designer selects both positive test scenario by taking valid input values and adverse test scenario by taking invalid input values to determine the correct output. Test cases are mainly designed for functional testing but can also be used for non-functional testing. Test cases are designed by the testing team, there is not any involvement of the development team of software.

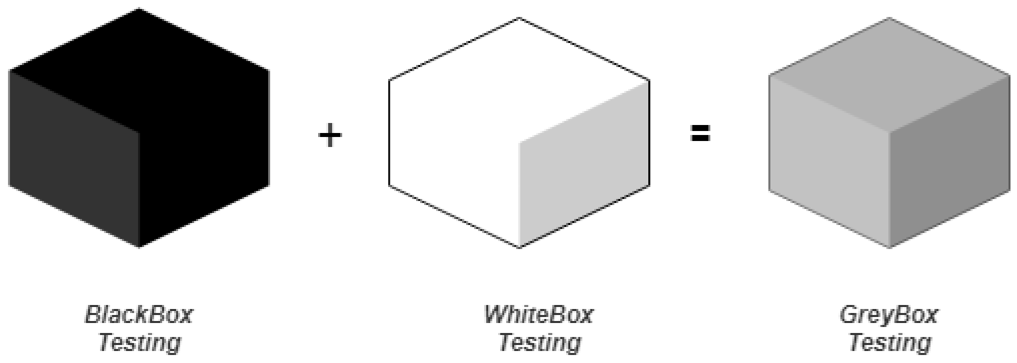
Techniques Used in Black Box Testing

Decision Table Technique	Decision Table Technique is a systematic approach where various input combinations and their respective system behavior are captured in a tabular form. It is appropriate for the functions that have a logical relationship between two and more than two inputs.
Boundary Value Technique	Boundary Value Technique is used to test boundary values, boundary values are those that contain the upper and lower limit of a variable. It tests, while entering boundary value whether the software is producing correct output or not.

State Transition Technique	State Transition Technique is used to capture the behavior of the software application when different input values are given to the same function. This applies to those types of applications that provide the specific number of attempts to access the application.
All-pair Testing Technique	All-pair testing Technique is used to test all the possible discrete combinations of values. This combinational method is used for testing the application that uses checkbox input, radio button input, list box, text box, etc.
Cause-Effect Technique	Cause-Effect Technique underlines the relationship between a given result and all the factors affecting the result.It is based on a collection of requirements.
Equivalence Partitioning Technique	Equivalence partitioning is a technique of software testing in which input data divided into partitions of valid and invalid values, and it is mandatory that all partitions must exhibit the same behavior.
Error Guessing Technique	Error guessing is a technique in which there is no specific method for identifying the error. It is based on the experience of the test analyst, where the tester uses the experience to guess the problematic areas of the software.
Use Case Technique	Use case Technique used to identify the test cases from the beginning to the end of the system as per the usage of the system. By using this technique, the test team creates a test scenario that can exercise the entire software based on the functionality of each function from start to end.

GreyBox Testing

Greybox testing is a software testing method to test the software application with partial knowledge of the internal working structure. It is a **combination of black box and white box testing** because it involves access to internal coding to design test cases as white box testing and testing practices are done at functionality level as black box testing.

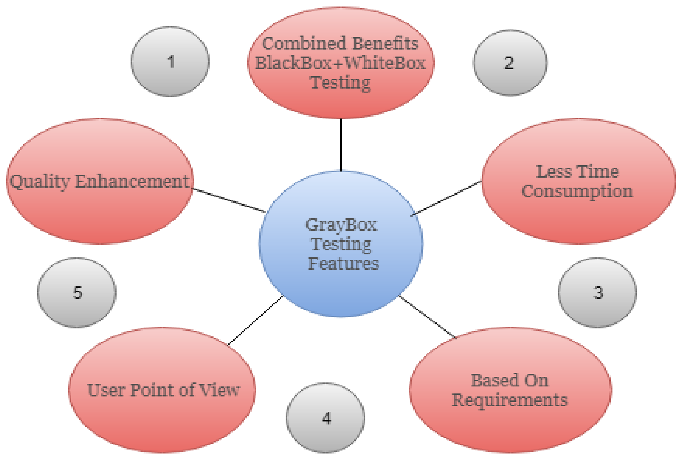


GreyBox testing commonly identifies context-specific errors that belong to web systems. For example; while testing, if tester encounters any defect then he makes changes in code to resolve the defect and then test it again in real time. It concentrates on all the layers of any complex software system to increase testing coverage. It gives the ability to test both presentation layer as well as internal coding structure. It is primarily used in integration testing and penetration testing.

Why GreyBox testing?

Reasons for GreyBox testing are as follows

- It provides combined benefits of both Blackbox testing and WhiteBox testing.
- It includes the input values of both developers and testers at the same time to improve the overall quality of the product.
- It reduces time consumption of long process of functional and non-functional testing.
- It gives sufficient time to the developer to fix the product defects.
- It includes user point of view rather than designer or tester point of view.
- It involves examination of requirements and determination of specifications by user point of view deeply.



GreyBox Testing Strategy

Grey box testing does not make necessary that the tester must design test cases from source code. To perform this testing test cases can be designed on the base of, knowledge of architectures, algorithm, internal states or other high -level descriptions of the program behavior. It uses all the straightforward techniques of black box testing for function testing. The test case generation is based on requirements and preset all the conditions before testing the program by assertion method.

Generic Steps to perform Grey box Testing are:

1. First, select and identify inputs from BlackBox and WhiteBox testing inputs.
2. Second, Identify expected outputs from these selected inputs.
3. Third, identify all the major paths to traverse through during the testing period.
4. The fourth task is to identify sub-functions which are the part of main functions to perform deep level testing.
5. The fifth task is to identify inputs for subfunctions.
6. The sixth task is to identify expected outputs for subfunctions.
7. The seventh task includes executing a test case for Subfunctions.
8. The eighth task includes verification of the correctness of result.

The test cases designed for Greybox testing includes Security related, Browser related, GUI related, Operational system related and Database related testing.

Techniques of Grey box Testing

Matrix Testing

This testing technique comes under Grey Box testing. It defines all the used variables of a particular program. In any program, variable are the elements through which values can travel inside the program. It should be as per requirement otherwise, it will reduce the readability of the program and speed of the software. Matrix technique is a method to remove unused and uninitialized variables by identifying used variables from the program.

Regression Testing

Regression testing is used to verify that modification in any part of software has not caused any adverse or unintended side effect in any other part of the software. During confirmation testing, any defect got fixed, and that part of software started working as intended, but there might be a possibility that fixed defect may have introduced a different defect somewhere else in the software. So, regression testing takes care of these type of defects by testing strategies like retest risky use cases, retest within a firewall, retest all, etc.

Pattern Testing

Pattern testing is applicable to such type of software that is developed by following the same pattern of previous software. In these type of software possibility to occur the same type of defects. Pattern testing determines reasons of the failure so they can be fixed in the next software.

Usually, automated software testing tools are used in Greybox methodology to conduct the test process. Stubs and module drivers provided to a tester to relieve from manually code generation.

Black Box Testing vs. White Box Testing vs. Grey Box Testing

Index	Black Box Testing	White Box Testing	Grey Box Testing
1	Knowledge of internal working structure (Code) is not required for this type of testing. Only GUI (Graphical User Interface) is required for test cases.	Knowledge of internal working structure (Coding of software) is necessarily required for this type of testing.	Partially Knowledge of the internal working structure is required.
2	Black Box Testing is also known as functional testing, data-driven testing, and closed box testing.	White Box Testing is also known as structural testing, clear box testing, code-based testing, and transparent testing.	Grey Box Testing is also known as translucent testing as the tester has limited knowledge of coding.
3	The approach towards testing includes trial techniques and error guessing method because tester does not need knowledge of internal coding of the software.	White Box Testing is proceeded by verifying the system boundaries and data domains inherent in the software as there is no lack of internal coding knowledge.	If the tester has knowledge of coding, then it is proceeded by validating data domains and internal system boundaries of the software.
4	The testing space of tables for inputs (inputs to be used for creating test cases) is pretty huge and largest among all testing spaces.	The testing space of tables for inputs (inputs to be used for creating test cases) is less as compared to Black Box testing.	The testing space of tables for inputs (inputs to be used for creating test cases) is smaller than Black Box and White Box testing.
5	It is very difficult to discover hidden errors of the software because errors can be due to internal working which is unknown for Black Box testing.	It is simple to discover hidden errors because it can be due to internal working which is deeply explored in White Box testing.	Difficult to discover the hidden error. Might be found in user level testing.

6	It is not considered for algorithm testing.	It is well suitable and recommended for algorithm testing.	It is not considered for algorithm testing.
7	Time consumption in Black Box testing depends upon the availability of the functional specifications.	White Box testing takes a long time to design test cases due to lengthy code.	Test cases designing can be done in a short time period.
8	Tester, developer and the end user can be the part of testing.	Only tester and developer can be a part of testing; the end user can not involve.	Tester, developer and the end user can be the part of testing.
9	It is the least time-consuming process among all the testing processes.	The entire testing process is the most time consuming among all the testing processes.	less time consuming than White Box testing.
10	Resilience and security against viral attacks are covered under Black Box testing.	Resilience and security against viral attacks are not covered under White Box testing.	Resilience and security against viral attacks are not covered under Grey Box testing.
11	The base of this testing is external expectations internal behavior is unknown.	The base of this testing is coding which is responsible for internal working.	Testing based on high-level database diagrams and dataflow diagrams.
12	It is less exhaustive than White Box and Grey Box testing methods.	It is most exhaustive between Black Box and Grey Box testing methods.	Partly exhaustive; depends upon the type of test cases are coding based or GUI based.