Black Box Testing is also known as behavioral, opaque-box, closed-box, specification-based or eye-to-eye testing.

It is a Software Testing method that analyzes the functionality of a software/application without knowing much about the internal structure/design of the item that is being tested and compares the input value with the output value.

**The main focus of Black Box Testing is on the functionality of the system as a whole.** The term **‘Behavioral Testing’** is also used for Black Box Testing.

Behavioral test design is slightly different from the black-box test design because the use of internal knowledge isn’t strictly forbidden, but it’s still discouraged. Each testing method has its own advantages and disadvantages. There are some bugs that cannot be found using black box or white box technique alone.

A majority of the applications are tested using the Black Box method. We need to cover the majority of test cases so that most of the bugs will get discovered by the Black-Box method.

This testing occurs throughout the Software Development and Testing Life Cycle i.e in Unit, Integration, System, Acceptance, and Regression Testing stages.

This can be either Functional or Non-Functional.

**Types of Black Box Testing**

Practically, there are several types of Black Box Testing that are possible, but if we consider a major variant of it then only the below mentioned are the two fundamental ones.

#1) Functional Testing

This testing type deals with the functional requirements or specifications of an application. Here, different actions or functions of the system are being tested by providing the input and comparing the actual output with the expected output.

**For example**, when we test a Dropdown list, we click on it and verify if it expands and all the expected values are showing in the list.

**Few major types of Functional Testing are:**

* Smoke Testing
* Sanity Testing
* Integration Testing
* System Testing
* Regression Testing
* User Acceptance Testing

#2) Non-Functional Testing

Apart from the functionalities of the requirements, there are even several non-functional aspects that are required to be tested to improve the quality and performance of the application.

**Few major types of Non-Functional Testing include:**

* Usability Testing
* Load Testing
* Performance Testing
* Compatibility Testing
* Stress Testing
* Scalability Testing

**Black Box Testing Techniques**

In order to systematically test a set of functions, it is necessary to design test cases. Testers can create test cases from the requirement specification document using the following Black Box Testing techniques:

* Equivalence Partitioning
* Boundary Value Analysis
* Decision Table Testing
* State Transition Testing
* Error Guessing
* Graph-Based Testing Methods
* Comparison Testing

**Advantages and Disadvantages**

**Advantages**

* The tester does not need to have a technical background. It is important to test by being in the user’s shoes and think from the user’s point of view.
* Testing can start once the development of the project/application is done. Both the testers and developers work independently without interfering in each other’s space.
* It is more effective for large and complex applications.
* Defects and inconsistencies can be identified in the early stages of testing.

**Disadvantages**

* Without any technical or programming knowledge, there are chances of ignoring possible conditions of the scenario to be tested.
* In a stipulated time there is a possibility of testing less and skipping all possible inputs and their output testing.
* Complete Test Coverage is not possible for large and complex projects.

## Conclusion

These are some of the basic points regarding Black box testing and the overview of its techniques and methods.

As it is not possible to test everything with human involvement with 100 percent accuracy, if the above-mentioned techniques and methods are used effectively, then it will definitely improve the quality of the system.

To conclude, this is a very helpful method to verify the functionality of the system and identify most of the defects.

**White box testing.** (also known as clear, glass box or structural testing) is a testing technique which evaluates the code and the internal structure of a program.

White box testing involves looking at the structure of the code. When you know the internal structure of a product, tests can be conducted to ensure that the internal operations performed according to the specification. And all internal components have been adequately exercised.

Why we perform WBT?

**To ensure:**

* That all independent paths within a module have been exercised at least once.
* All logical decisions verified on their true and false values.
* All loops executed at their boundaries and within their operational bounds internal data structures validity.

**To discover the following types of bugs:**

* Logical error tend to creep into our work when we design and implement functions, conditions or controls that are out of the program
* The design errors due to difference between logical flow of the program and the actual implementation
* Typographical errors and syntax checking

Does this testing requires detailed programming skills?

We need to write [test cases](https://www.softwaretestinghelp.com/how-to-write-effective-test-cases-test-cases-procedures-and-definitions/) that ensure the complete coverage of the program logic.

For this we need to know the program well i.e. We should know the specification and the code to be tested. Knowledge of programming languages and logic is required for this type of testing.

Limitations

Not possible for testing each and every path of the loops in the program. This means exhaustive testing is impossible for large systems.

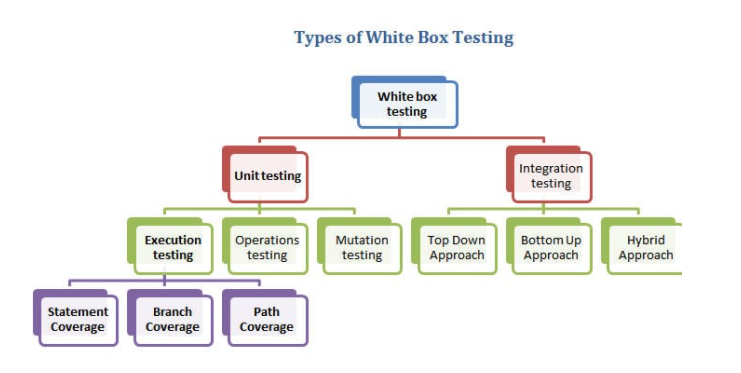
This does not mean that WBT is not effective. By selecting important logical paths and data structure for testing is practically possible and effective.

### Steps to Perform WBT

**Step #1** – Understand the functionality of an application through its source code. Which means that a tester must be well versed with the programming language and the other tools as well techniques used to develop the software.

**Step #2**– Create the tests and execute them.

When we discuss the concept of testing, “[coverage](https://www.softwaretestinghelp.com/test-coverage/)” is considered to be the most important factor. Here I will explain how to have maximum coverage from the context of White box testing.



**3 Main White Box Testing Techniques:**

1. Statement Coverage
2. Branch Coverage
3. Path Coverage

Note that the statement, branch or path coverage does not identify any bug or defect that needs to be fixed. It only identifies those lines of code which are either never executed or remains untouched. Based on this further testing can be focused on.

Let’s understand these techniques one by one with a simple example.

**#1) Statement coverage:**

In a programming language, a statement is nothing but the line of code or instruction for the computer to understand and act accordingly. A statement becomes an executable statement when it gets compiled and converted into the object code and performs the action when the program is in a running mode.

Hence *“Statement Coverage”*, as the name itself suggests, it is the method of validating whether each and every line of the code is executed at least once.

**#2) Branch Coverage:**

“Branch” in a programming language is like the “IF statements”. An IF statement has two branches: T**rue and False**.

So in Branch coverage (also called Decision coverage), we validate whether each branch is executed at least once.

**In case of an “IF statement”, there will be two test conditions:**

* One to validate the true branch and,
* Other to validate the false branch.

Hence, in theory, Branch Coverage is a testing method which is when executed ensures that each and every branch from each decision point is executed.

**#3) Path Coverage**

Path coverage tests all the paths of the program. This is a comprehensive technique which ensures that all the paths of the program are traversed at least once. Path Coverage is even more powerful than Branch coverage. This technique is useful for testing the complex programs.

### White Box Testing Tools

* **Veracode**
* **Googletest**
* **EclEmma**
* **JUnit**

JUnit is a quiet simple unit testing framework that supports test automation in Java Programming Language. It mainly supports in Test Driven Development and provides the Test coverage report as well. It is licensed under Eclipse Public License. For free download and in order to know more about JUnit please check the below link.

### ****Conclusion****

Relying only on black box testing is not sufficient for maximum test coverage. We need to have a combination of both black box and white box testing techniques to [cover maximum defects](https://www.softwaretestinghelp.com/tips-to-find-valid-defects-in-any-application/).

If done properly, White box testing will certainly contribute to the software quality. It’s also good for testers to participate in this testing as it can provide the most “unbiased” opinion about the code. :)

| **Black Box Testing** | **White Box Testing** |
| --- | --- |
| It is a testing method without having knowledge about the actual code or internal structure of the application. | It is a testing method having knowledge about the actual code and internal structure of the application. |
| This is a higher level testing such as functional testing. | This type of testing is performed at a lower level of testing such as Unit Testing, Integration Testing. |
| It concentrates on the functionality of the system under test. | It concentrates on the actual code – program and its syntax's. |
| Black box testing requires Requirement specification to test. | White Box testing requires Design documents with data flow diagrams, flowcharts etc. |
| Black box testing is done by the testers. | White box testing is done by Developers or testers with programming knowledge. |

**Grey Box Testing** or Gray box testing is a software testing technique to test a software product or application with partial knowledge of internal structure of the application. The purpose of grey box testing is to search and identify the defects due to improper code structure or improper use of applications.

In this process, context-specific errors that are related to web systems are commonly identified. It increases the testing coverage by concentrating on all of the layers of any complex system.

Gray Box Testing is a software testing method, which is a combination of both [White Box Testing](https://www.guru99.com/white-box-testing.html) and Black Box Testing method.

In Grey Box Testing internal structure (code) is partially known

## **Why Gray Box Testingx**

Gray Box Testing is performed for the following reason,

* It provides combined benefits of both black box testing and white box testing both
* It combines the input of developers as well as testers and improves overall product quality
* It reduces the overhead of long process of testing functional and non-functional types
* It gives enough free time for a developer to fix defects
* Testing is done from the user point of view rather than a designer point of view

## **Gray Box Testing Strategy**

To perform Gray box testing, it is not necessary that the tester has the access to the source code. A test is designed based on the knowledge of algorithm, architectures, internal states, or other high -level descriptions of the program behavior.

To perform Gray box Testing-

* It applies a straightforward technique of black box testing
* It is based on requirement test case generation, as such, it presets all the conditions before the program is tested by assertion method.

##### Techniques used for Grey box Testing are-

* **Matrix Testing:**This testing technique involves defining all the variables that exist in their programs.
* **Regression Testing**: To check whether the change in the previous version has regressed other aspects of the program in the new version. It will be done by testing strategies like retest all, retest risky use cases, retest within a firewall.
* **Orthogonal Array Testing or OAT**: It provides maximum code coverage with minimum test cases.
* **Pattern Testing:** This testing is performed on the historical data of the previous system defects. Unlike black box testing, gray box testing digs within the code and determines why the failure happened

Usually, Grey box methodology uses automated software testing tools to conduct the testing. Stubs and module drivers are created to relieve tester to manually generate the code.

##### **Steps to perform Grey box Testing are:**

* **Step 1**: Identify inputs
* **Step 2**: Identify the outputs
* **Step 3**: Identify the major paths
* **Step 4**: Identify Subfunctions
* **Step 5**: Develop inputs for Subfunctions
* **Step 6**: Develop outputs for Subfunctions
* **Step 7**: Execute test case for Subfunctions
* **Step 8**: Verify the correct result for Subfunctions
* **Step 9**: Repeat steps 4 & 8 for other Subfunctions
* **Step 10**: Repeat steps 7 & 8 for other Subfunctions

The test cases for grey box testing may include, GUI related, Security related, Database related, Browser related, Operational system related, etc.

## Gray Box Testing Challenges

* When a component under test encounter a failure of some kind may lead to abortion of the ongoing operation
* When test executes in full but the content of the result is incorrect.

**Summary:**

* The overall cost of system defects can be reduced and prevented from passing further with Grey box testing
* Grey box testing is suited more for GUI,[Functional Testing](https://www.guru99.com/functional-testing.html), security assessment, web applications, web-services, etc.
* Techniques used for Grey box Testing
  + Matrix Testing
  + [Regression Testing](https://www.guru99.com/regression-testing.html)
  + OAT or Orthogonal Array Testing
  + Pattern Testing