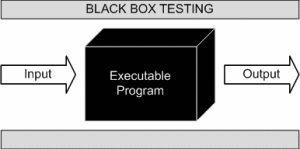
Black Box Testing

**BLACK BOX TESTING**, also known as Behavioral Testing, is a [software testing method](http://softwaretestingfundamentals.com/software-testing-methods/) in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.



This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

• Incorrect or missing functions

• Interface errors

• Errors in data structures or external database access

• Behavior or performance errors

• Initialization and termination errors

Definition by ISTQB

**• 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.

Example

A tester, without knowledge of the internal structures of a website, tests the web pages by using a browser; providing inputs (clicks, keystrokes) and verifying the outputs against the expected outcome.

Levels Applicable To

Black Box Testing method is applicable to the following levels of software testing:

• [Integration Testing](http://softwaretestingfundamentals.com/integration-testing/)

• [System Testing](http://softwaretestingfundamentals.com/system-testing/)

• [Acceptance Testing](http://softwaretestingfundamentals.com/acceptance-testing/)

The higher the level, and hence the bigger and more complex the box, the more black-box testing method comes into use.

Techniques

Following are some techniques that can be used for designing black box tests.

*• Equivalence Partitioning:*It is a software test design technique that involves dividing input values into valid and invalid partitions and selecting representative values from each partition as test data.

*• Boundary Value Analysis:*It is a software test design technique that involves the determination of boundaries for input values and selecting values that are at the boundaries and just inside/ outside of the boundaries as test data.

*• Cause-Effect Graphing:*It is a software test design technique that involves identifying the cases (input conditions) and effects (output conditions), producing a Cause-Effect Graph, and generating test cases accordingly.

Advantages

• Tests are done from a user’s point of view and will help in exposing discrepancies in the specifications.

• Tester need not know programming languages or how the software has been implemented.

• Tests can be conducted by a body independent from the developers, allowing for an objective perspective and the avoidance of developer-bias.

• Test cases can be designed as soon as the specifications are complete.

Disadvantages

• Only a small number of possible inputs can be tested and many program paths will be left untested.

• Without clear specifications, which is the situation in many projects, test cases will be difficult to design.

• Tests can be redundant if the software designer/developer has already run a test case.

• Ever wondered why a soothsayer closes the eyes when foretelling events? So is almost the case in Black Box Testing.

Black Box Testing is contrasted with [White Box Testing](http://softwaretestingfundamentals.com/white-box-testing/). Read [Differences between Black Box Testing and White Box Testing.](http://softwaretestingfundamentals.com/differences-between-black-box-testing-and-white-box-testing/)