# Accessors and Modifiers Encyclopedia Article

## Accessors and Modifiers

The following sections of this BookRags Literature Study Guide is offprint from Gale's For Students Series: Presenting Analysis, Context, and Criticism on Commonly Studied Works: Introduction, Author Biography, Plot Summary, Characters, Themes, Style, Historical Context, Critical Overview, Criticism and Critical Essays, Media Adaptations, Topics for Further Study, Compare & Contrast, What Do I Read Next?, For Further Study, and Sources.

(c)1998-2002; (c)2002 by Gale. Gale is an imprint of The Gale Group, Inc., a division of Thomson Learning, Inc. Gale and Design and Thomson Learning are trademarks used herein under license.

The following sections, if they exist, are offprint from Beacham's Encyclopedia of Popular Fiction: "Social Concerns", "Thematic Overview", "Techniques", "Literary Precedents", "Key Questions", "Related Titles", "Adaptations", "Related Web Sites". (c)1994-2005, by Walton Beacham.

The following sections, if they exist, are offprint from Beacham's Guide to Literature for Young Adults: "About the Author", "Overview", "Setting", "Literary Qualities", "Social Sensitivity", "Topics for Discussion", "Ideas for Reports and Papers". (c)1994-2005, by Walton Beacham.

All other sections in this Literature Study Guide are owned and copyrighted by BookRags, Inc.

# Contents

Accessors and Modifiers Encyclopedia Article.....1

Contents.....2

Accessors and Modifiers.....3

# Accessors and Modifiers

Accessors and modifiers are operations in **object-oriented programming** that read or modify some property of an **object**. Accordingly, operations that function as accessors and modifiers are designed to read or modify variables in a software bundle that generally affect the state and behavior of an object.

Accessors do not change the state of the object. Read **access** is an example of an accessor **operation**. An accessor **method** returns, or displays, the **value** of a **data member**. Accessors do not use parameters and they are designed to remove the need for client programs to directly access **data** or to provide read-only access to data. Accessors representing values take the place of the data in calculation or output operations.

Modifiers allow changes to the state of an object **variable**. Modifications are, however, designed to prevent changes in data relationships (maintain integrity) by a particular function. Such protection is important because changes to the value of a data member are often critical. For example, if a **bug** is introduced during the modification operation, than the data can become corrupted and the program can cease to function. There are many types of such corruption errors, but in common among such errors is a loss of continuity in either the reading of program instructions or an introduced corruption of data type (a change in data format that is incompatible with the operation). In contrast to accessors, modifiers do not return values.

Accessors and modifiers are both **member function** types, along with constructors and destructors. Generally, they permit the sole direct access by a user to a **class**.