Base Class Encyclopedia Article

Base Class

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

Base Class Encyclopedia Article.....1

Contents.....2

Base Class.....3

Base Class

A base **class** (also called superclass or **parent** class) is a term from object-oriented **design** that denotes a class that another class extends--other classes are based on it. In other words, if class B is based on class A, then class A is said to be the base class of class B. Programmatically this might look something like this in C++:

- class Haircut {}; // "base class" or "superclass"
- class Mohawk : public Haircut {};

and like this in Java:

- class Haircut {}; // "base class" or "superclass"
- class Mohawk extends Haircut {};

If Mohawk were to be extended once more by deriving another class from it, then it might look like this:

- class ColouredMohawk : public Mohawk {} // in C++

or

- class ColouredMohawk extends Mohawk {} // in Java

In each of these cases Mohawk becomes the base class of ColouredMohawk.

The opposite of a base class is a "derived class" (or "subclass"). Thus, if Haircut is the base class of Mohawk, then Mohawk is a **derived class** of Haircut.

The relationship between a derived class and its base class is sometimes called a "kind of" relationship, where the derived class is a "kind of" its base class. Using the examples above it is easy to see that a Mohawk is a "kind of" haircut, but the reverse is rather nonsensical (a haircut is not a "kind of" Mohawk).

When classes are arranged like this in terms of base class" and derived class it is called a "class hierarchy," with the more general classes, like Haircut, at the top and the more specific classes, like ColouredMohawk at the bottom.

Derived classes typically "inherit" the **data** and functionality of their superclasses (though exactly how depends on the language used) and then add more functionality and data of their own to fulfil their own requirements. In general a derived class is more complex and specific in terms of functionality and the data it comprises than its base class.

There is sometimes a subtle difference in meaning between the expressions "base class" and "superclass." The expression "base class" can be used to denote the very topmost class of the hierarchy as well as a base class of a specific class, whereas the

expression "superclass" is generally used as a more relative term to mean "the base class of this class"; however, the terms are almost completely interchangeable and only a purist would worry about the differences.