s

scope The spatial or textual region of a program or form within which it is possible to refer to an object.

standard error A predefined I/O stream used to alert the user to some exceptional condition in the program during execution.

symbol An object used as an identifier to specify a relationship between a name and other objects and variables. Internally, each symbol is represented as a structure with the following components:

- Value
- Property list

These contain information about the symbol, and functions are provided to manipulate this information, as well as the symbol itself.

t

template Provides a means of defining a complex macro that supports parameterized classes.

typedef A storage class that is used to create new data types from existing data types.

u

unordered sequence class A collection of basic data structures that implement random-access data structures as parameterized classes.

v

value A constant or quantity assigned to a variable.

void A data type that is used when declaring a function to indicate that the function does not return a value or that the function does not take any arguments.

o

object A variable declared to be of a specific class. An object is not just passive data, but also the procedures which manipulate it. Objects are the modular building blocks for an object-oriented programming system.

object-oriented programming A programming approach for designing and implementing software systems, centering around the concepts of abstract data types and classes, hierarchies, inheritance, and polymorphism. Noted primarily for its advantages of code reuse, extensibility, complexity control, and much closer linkage between software design and implementation.

operator A symbol specifying an arithmetic, logical, or other manipulation of its operands.

ordered sequence class A collection of basic data structures that implement sequential-access data structures as parameterized classes.

overloaded operator An operator with an additional meaning assigned to it. When an operator is overloaded, its meaning is usually inferred from the types of their operands.

p

package A collection of symbols that serves as a namespace. See also package system.

package system A facility that establishes a mapping from names to symbols and helps prevent namespace conflicts. The package system allows different programs to use the same name for objects so that the programs and objects can coexist in the same environment.

parameterized class A class in which one or more types can be declared at compile time.

pointer A data type that holds the address of an object in memory.

polymorphic The ability of different objects to respond differently to the same message at run time.

private Information that cannot be manipulated by the programmer.

procedures The operations or behaviors that the object can perform.

property list A component of a symbol that effectively provides the symbol with many modifiable, named components. The property list has zero or more entries, with each entry consisting of a pair of elements. The first element of the pair is called the indicator and is used to name a particular property. Each indicator must be unique within that property list. The second element can be any object that represents the value of that property. Functions are available tomanipulate a symbol's property list.

protected Information that can be manipulated in a limited manner by the programmer.

public Information that can be manipulated by the programmer.

r

raise To announce an exception.

h

hash table A table that derives a numeric index from some data key to index a specific value.

header file A file containing information needed by several program modules. During compilation, the text of the header file becomes part of the program text that the compiler analyzes.

heterogeneous The condition in which several objects are of different types.

homogeneous The condition in which several objects are of the same type.

i

IMPLEMENT A macro that expands into the function definitions of a class.

#include A simple text manipulation mechanism for gathering source program fragments together into a single file for compilation.

inheritance 1. The ability of a class to use properties of another class. Enables the programmer to define an organization of classes that models the relationships among the various kinds of objects.

2. The capability for distinguishing between the generic properties of some class of object and the more specialized properties that only certain objects will share.

interned symbol A symbol that belongs to a specific package.

iterator A mechanism that automatically repeats the same series of steps until a predetermined stop is reached.

m

macro A simple, symbolic programming-language statement that, when expanded, results in a series of more complex statements.

member function The set of operations defined to manipulate an object within a class.

n

NULL An empty or nonexistent or non-specified value.

derived class A class that inherits from a base class.

destructors Member functions providing for the automatic deallocation of the storage occupied by an object when the block containing the object is exited.

dymanic enumeration type A modifiable data type used to define a set of named integral constants and to declare variables of that type.

e

encapsulation A special type of form that surrounds another form and enhances the other form's operation without changing its basic functionality. A trace, for example, is an encapsulation.

enum Keyword for declaring an enumeration.

enumeration A set of symbolic integral constants.

environmental synonym A variable contained in the operating system environment in which an application program runs and provides a specific value or customization directive to the program.

exception Some noteworthy event that can occur during the execution of a program, such as an error or anomaly.

exception handling A mechanism for managing program anomalies and errors.

extensibility A language feature that allows programmers to create new types that can be endowed with specific properties and whose behavior is characterized in a class definition.

f

friend A nonmember of a class that is given access to the nonpublic members of the class. Can be a nonmember function, a member function, or an entire other class.

g

Generic class The class that is inherited by most COOL classes. It is used as a base class that adds run-time type checking and basic print capabilities to any derived class.

GLOSSARY

a

abstract data type A set of values and a set of operations that can be applied to the new type.

accessor 1. A trivial inline function that gets or sets the value of a private or protected slot of a class.

2. A function designed to access (that is, read, write, or modify) the value(s) of a private or protected slot of a class.

array A collection of objects of a single data type.

assignment statement A programming language statement that gives a value to a variable.

association list A data structure consisting of a list of pairs where each pair represents an association between its objects. The first element of the pair is the key and the second element is associated data. This is also referred to as an *alist*.

b

base class The class from which subclasses are derived and inherit their properties.

c

class The basic building blocks of an application, where each individual aspect of structure and behavior is defined separately.

constructors Member functions with the same name as the class in which they are defined that provide for the automatic initialization of objects at their point of declaration.

container class A class such as **Vector**, **List**, and **Hash_Table** that contains a set of application programmer defined data types. The COOL library contains a number of container classes.

d

DECLARE A macro that expands to the declaration of a class.

#define Preprocessor directive that defines a name and (optionally) the value that follows it.

derivation The process by which one class is built on top of (specialized) from one or more base classes. For example, the **Bit_Set** class is derived from the **Generic** class, whereas the **Generic** class is the base class of **Bit_Set**.

Printed on: Wed Apr 18 07:19:30 1990

Last saved on: Tue Apr 17 12:46:10 1990

Document: glossary

For: skc