Glossary: Advanced Object-Oriented Programming Concepts

Estimated time: 3 minutes

Welcome! This alphabetized glossary contains many terms used in this module. Understanding these terms is essential when working in the industry, participating in user groups, and participating in other certificate programs.

Term	Definition
Abstract classes	Contain abstract methods that must be implemented by subclasses and concrete methods that can be used or overridden by subclasses.
Abstraction	Simplifies complex systems by hiding unnecessary details and exposing only the essential features.
Anonymous inner class	Does not have a name and is used to instantiate a class that may not be reused elsewhere. Anonymous inner classes are often used to create a subclass, implement interfaces, or extend classes with minimal code.
Code reusability	Allowing a class to inherit methods and properties from another class, eliminating the need to rewrite existing code.
Compile-time polymorphism	Occurs when multiple methods belonging to the same class have the same name but differ in the number or type of parameters. It is also known as method overloading.
Decoupling	Separation of code that uses an interface from the code that implements it, resulting in loosely coupled systems that are easier to maintain.
Dynamic method resolution	Enables changing behavior at runtime and supports complex systems.
extends keyword	Used to define that a class is inheriting from a superclass, enabling code reuse and extension.
Hierarchical classification	Helps organize classes by having subclasses inherit from a common superclass, adding specific attributes.
Hierarchical inheritance	Enables multiple subclasses to inherit from a common superclass, making it efficient for shared functionality.
Inheritance	Concept in which one class, known as the child, derives properties and methods from another class, the parent.
Inner class	Defined within another class in Java. Helps organize code better.
Interface	Reference type that defines a contract for classes to follow. It can contain constants, method signatures, default methods, static methods, and nested types but cannot have instance fields or constructors.
Main class	Exhibits implementation of the abstract method and inherits concrete behavior from an abstract class.
Method-local inner class	Defined within a method of the outer class. Method-only local classes can only be accessed within that method.
Method overriding	Allows the child to perform specific tasks in its own way, making its behavior more flexible and dynamic.
Method signature	Allows only method declarations without bodies and provides a blueprint for implementing classes.
Multilevel inheritance	Allows inheritance from other subclasses.
Multiple inheritance	Indicates the process of one subclass inheriting properties from multiple superclasses.
Non-static inner class	Can access both static and non-static members of the outer class.
Organized code	Structures classes into hierarchies, making it easier to understand and extend functionality.
Polymorphism	Process that allows objects to share the same interface while exhibiting different behaviors based on their specific implementations.
Runtime polymorphism	Occurs when a subclass provides its specific implementation of a method defined in its superclass. It is also known as method overriding.
Single inheritance	Allows a subclass to inherit from just one superclass, simplifying the structure.
Static nested class	Cannot access non-static members of the outer class directly. Static nested classes are associated with the outer class but do not require an instance of the outer class for instantiation.
Subclass	Inherits properties and methods from the superclass.
Superclass	Class whose properties and methods are inherited by another class.
Virtual method	Any method that can be overridden in a subclass, enabling dynamic method resolution.

Skills Network