Most Common C++ Design Patterns for All Projects

Singleton

Ensures a class has only one instance and provides a global point of access to it.

Usage: Logging, configuration managers, resource managers.

C++ Note: Thread-safety must be considered.

Factory Method

Creates objects without specifying the exact class of the object that will be created.

Usage: When the exact type is determined at runtime.

Strategy

Allows selecting an algorithm's behavior at runtime.

Usage: Swappable sorting or rendering algorithms, game AI strategies.

Observer

Allows objects to be notified of state changes in other objects.

Usage: GUI frameworks, event systems, decoupled communications.

Command

Encapsulates a request as an object, allowing for parameterization and queuing of requests.

Usage: Undo/redo systems, input handling in games, task scheduling.

Adapter

Converts one interface to another expected by clients.

Usage: Integrating legacy code or different APIs.

Decorator

Adds behavior to objects dynamically without altering their structure.

Usage: UI components, enhancing I/O streams.

Template Method

Defines the skeleton of an algorithm, deferring some steps to subclasses.

Usage: Framework/library design, enforcing a sequence of operations.

Composite

Treats individual objects and compositions of objects uniformly.

Usage: Hierarchical data like GUI elements, scene graphs.

State

Allows an object to change its behavior when its internal state changes.

Usage: Game development (character states), workflow engines.