**Command Pattern**

The Command Pattern is a behavioral design pattern that transforms a request or an operation into an object, allowing for parameterization and queuing of requests, and providing additional functionalities like undo/redo operations and logging. It separates the sender (client) of a request from the receiver (object that performs the action), encapsulating the request as an object with a specific interface.

Key components of the Command Pattern:

**Command**: Interface or abstract class that declares the execute method. Each concrete command class implements this interface and encapsulates a specific action or operation.

**Concrete Command**: Classes that implement the Command interface by providing the specific implementation of the execute method. Each concrete command is associated with a receiver and defines how to perform a particular action.

**Receiver**: The object that performs the actual action associated with a command. It contains the business logic and knows how to carry out the operation requested by the command.

**Invoker**: The client or an object responsible for invoking a command. It holds a reference to a command and triggers the execution of the command when necessary.

**Client**: The component that creates and configures the concrete command objects, associates them with the appropriate receivers, and sets up the invokers to use the commands.

The flow of execution in the Command Pattern typically follows these steps:

The client creates a specific command object and associates it with a receiver.

The client sets the command object in an invoker.

The client triggers the command execution through the invoker.

The invoker invokes the execute method of the associated command.

The command delegates the action to the receiver for actual execution.

Advantages of using the Command Pattern include decoupling the sender from the receiver, allowing for easier extensibility and modification, enabling undo/redo operations, supporting queuing and logging of commands, and facilitating testing and reusability of components.