PareDox
Education Solutions

Aamir Shabbir Pare

Problem Solving Programming

Design Patterns

PareDox
Education Solutions

Previous Lecture

- Facade Design Pattern
- Click on Lecture – Facade Design Pattern

  OR

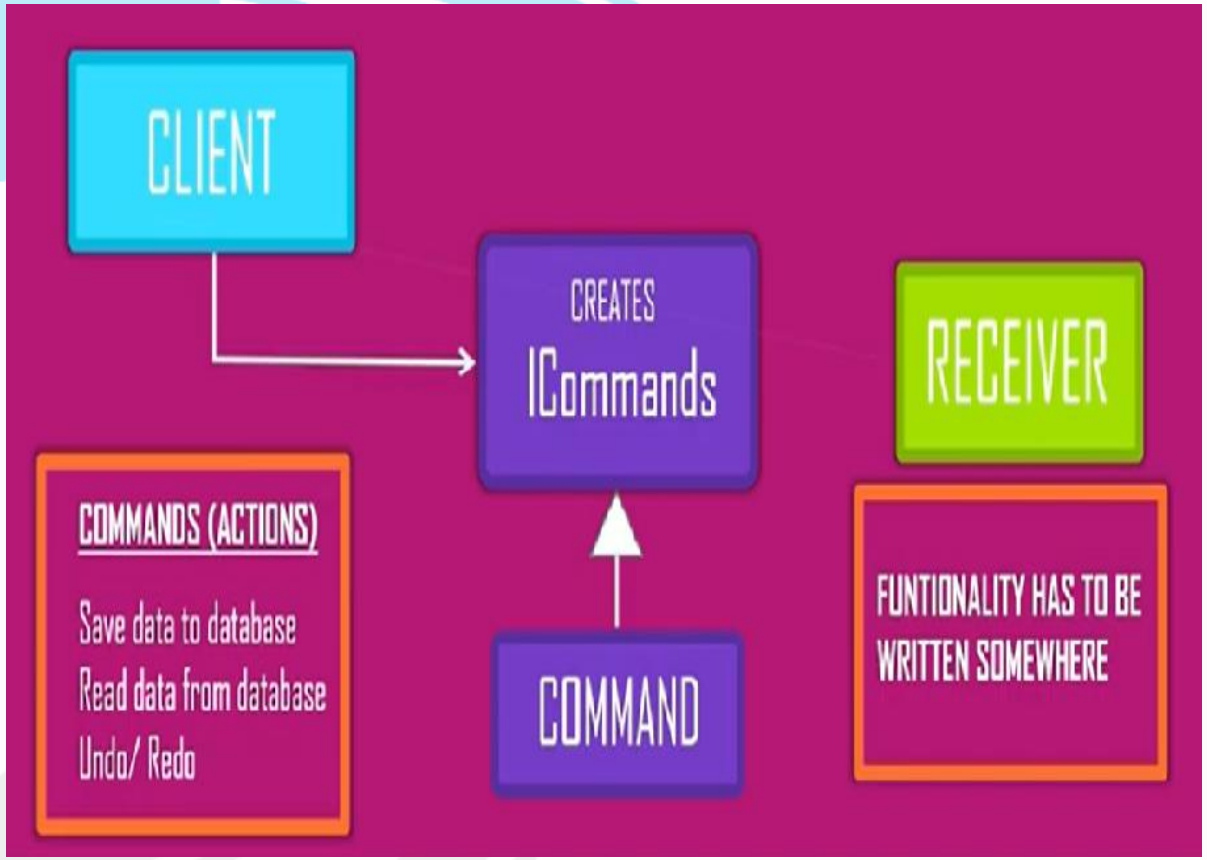
- Go to teams and watch video lecture

Aamir Shabbir Pare

PareDox
Education Solutions

Command Design Pattern

- Command is a behavioral design pattern that turns a request into a stand-alone object that contains all information about the request. This transformation lets you parameterize methods with different requests, delay or queue a request's execution, and support undoable operations.



Aamir Shabbir Pare

PareDox
Education Solutions

Structure

- Microsoft used this pattern in Windows Presentation Foundation (WPF). The article at https://visualstudiomagazine.com/articles/2012/04/10/command-pattern-innet.aspx describes it in detail.
- “The command pattern is well suited for handling GUI interactions. It works so well that Microsoft has integrated it tightly into the Windows Presentation Foundation (WPF) stack. The most important piece is the ICommand interface from the System.Windows.Input namespace. Any class that implements the ICommand interface can be used to handle a keyboard or mouse event through the common WPF controls. This linking can be done either in XAML or in a code-behind.

PareDox
Education Solutions

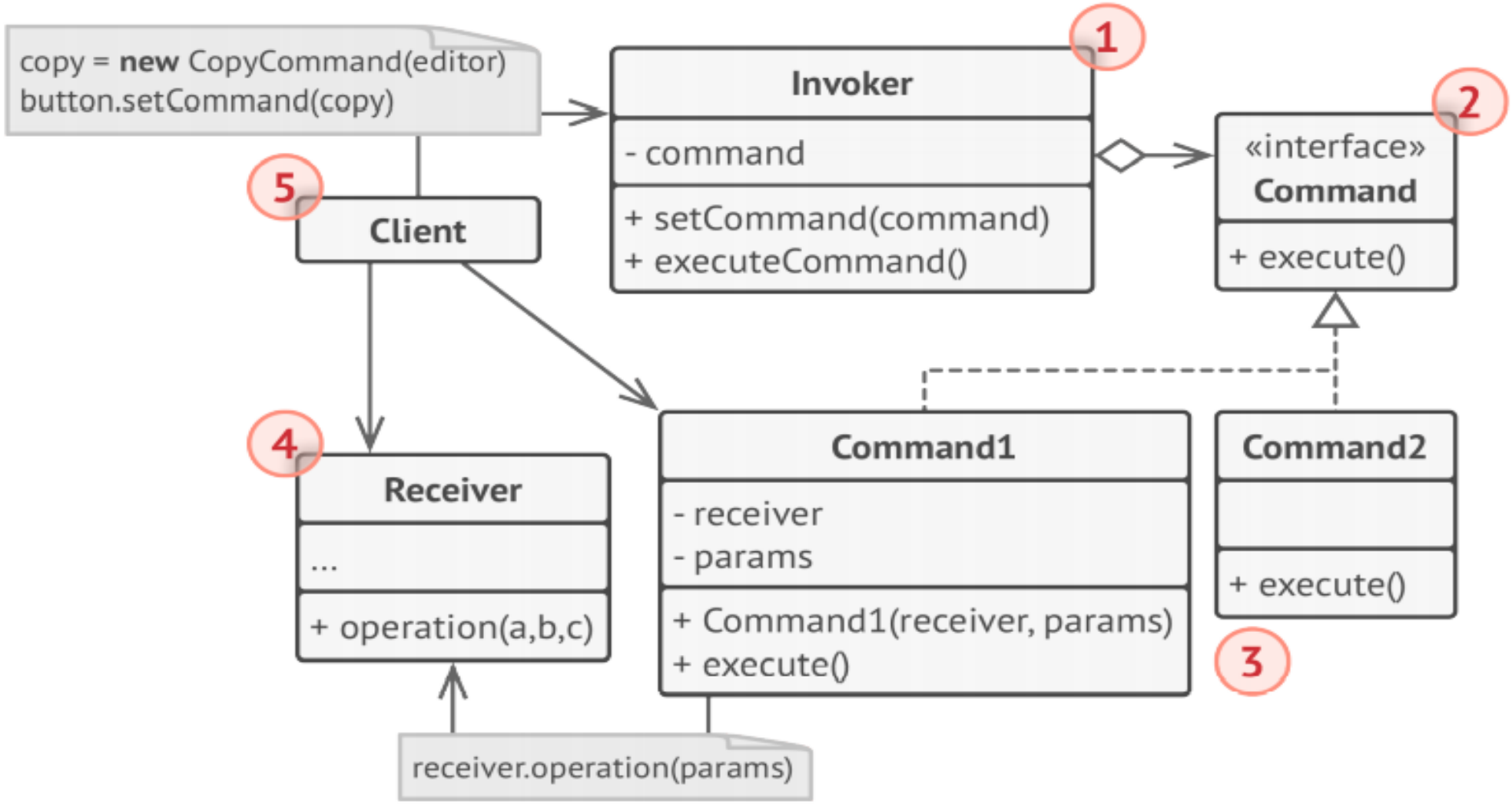
Command Concept

- The client only makes the decision (which commands to execute) and then passes the command to the invoker object (for the execution).
- Encapsulate a request as an object, thereby letting you parameterize clients with different requests, queue or log requests, and support undoable operations.
- With this pattern, you encapsulate a method invocation process.
- A command object can invoke a method of the receiver in a way that is specific to that receiver's class. The receiver starts processing the job.
- A command object is separately passed to the invoker object to invoke the command.
- The client object holds the invoker object and the command objects.
- The client only makes the decision (which commands to execute) and then passes the command to the invoker object (for the execution)
- Command pattern helps to track the history of executed operations and makes it possible to revert an operation if needed.

Aamir Shabbir Pare

PareDox
Education Solutions

Structure



Aamir Shabbir Pare

PareDox
Education Solutions

Structure Details

- The **Sender** class (aka invoker) is responsible for initiating requests. This class must have a field for storing a reference to a command object. The sender triggers that command instead of sending the request directly to the receiver. Note that the sender isn't responsible for creating the command object. Usually, it gets a pre-created command from the client via the constructor.
- The **Command** interface usually declares just a single method for executing the command.
- **Concrete Commands** implement various kinds of requests. A concrete command isn't supposed to perform the work on its own, but rather to pass the call to one of the business logic objects.
- However, for the sake of simplifying the code, these classes can be merged. Parameters required to execute a method on a receiving object can be declared as fields in the concrete command. You can make command objects immutable by only allowing the initialization of these fields via the constructor.

PareDox
Education Solutions

Structure Details

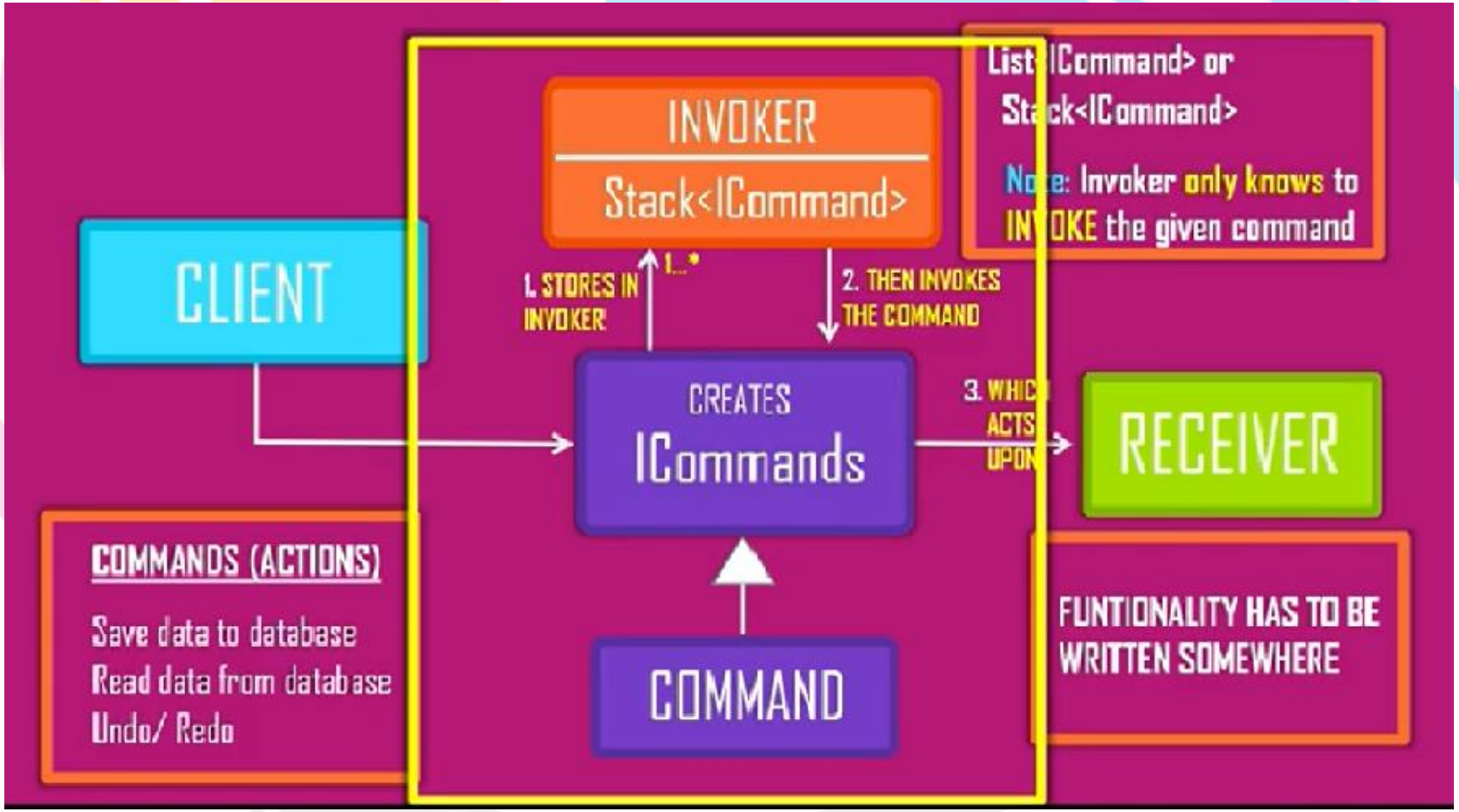
- The **Receiver** class contains some business logic. Almost any object may act as a receiver. Most commands only handle the details of how a request is passed to the receiver, while the receiver itself does the actual work.
- The **Client** creates and configures concrete command objects. The client must pass all of the request parameters, including a receiver instance, into the command's constructor. After that, the resulting command may be associated with one or multiple senders

Aamir Shabbir Pare

PareDox
Education Solutions

Core of Command Pattern

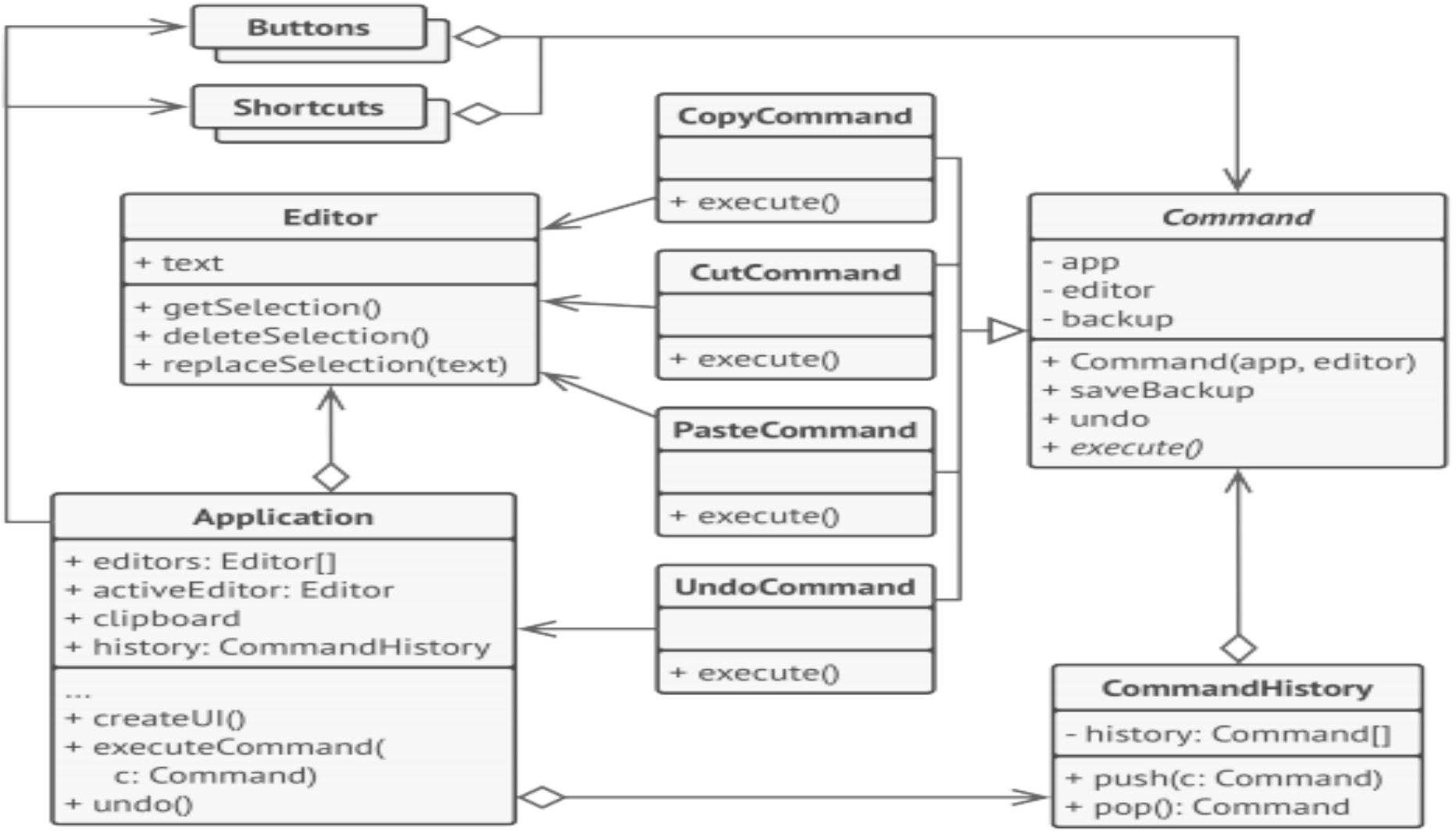
Invoker, ICommands and Command are at the core of this design pattern.



Aamir Shabbir Pare

PareDox
Education Solutions

Text Editor Example



Text Editor Operations

Aamir Shabbir Pare