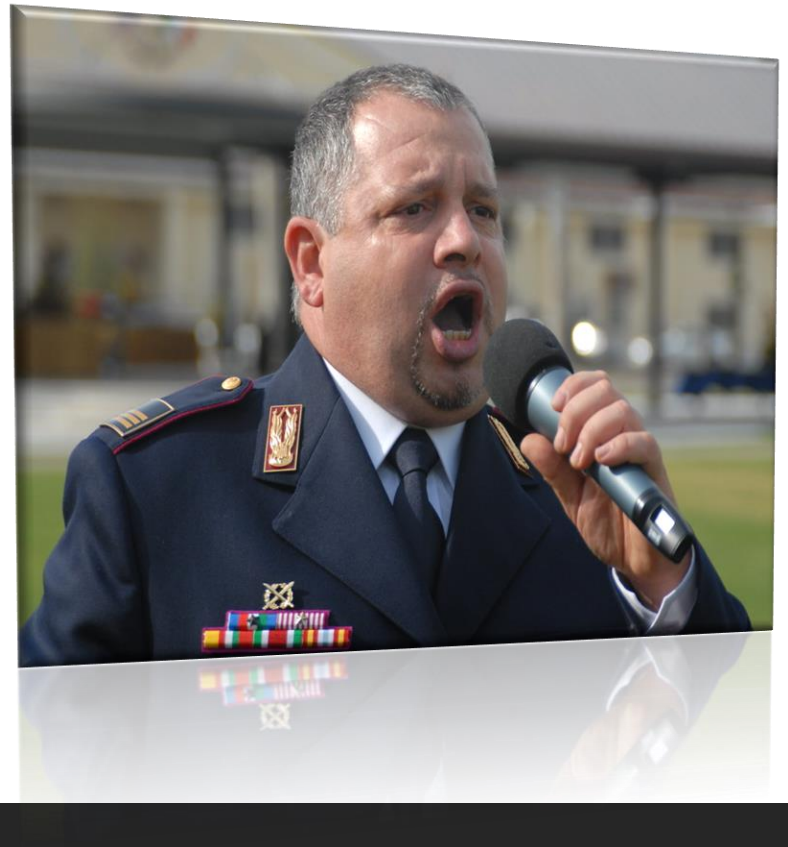
# Patrones de Diseño de Software

Patrones de Comportamiento - Command

By Luis Fernando Hurtado Loayza

# Patron de diseño Command

## PROPOSITO:



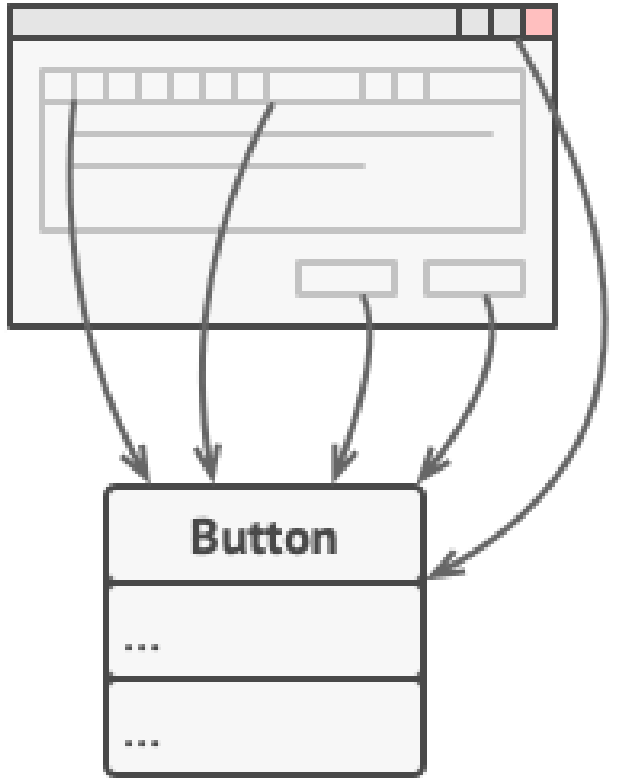
**Command** es un patrón de diseño de comportamiento que convierte una solicitud en un objeto independiente que contiene toda la información sobre la solicitud. Esta transformación nos permite parametrizar los métodos con diferentes solicitudes, retrasar o poner en cola la ejecución de una solicitud y soportar operaciones que no se pueden realizar.

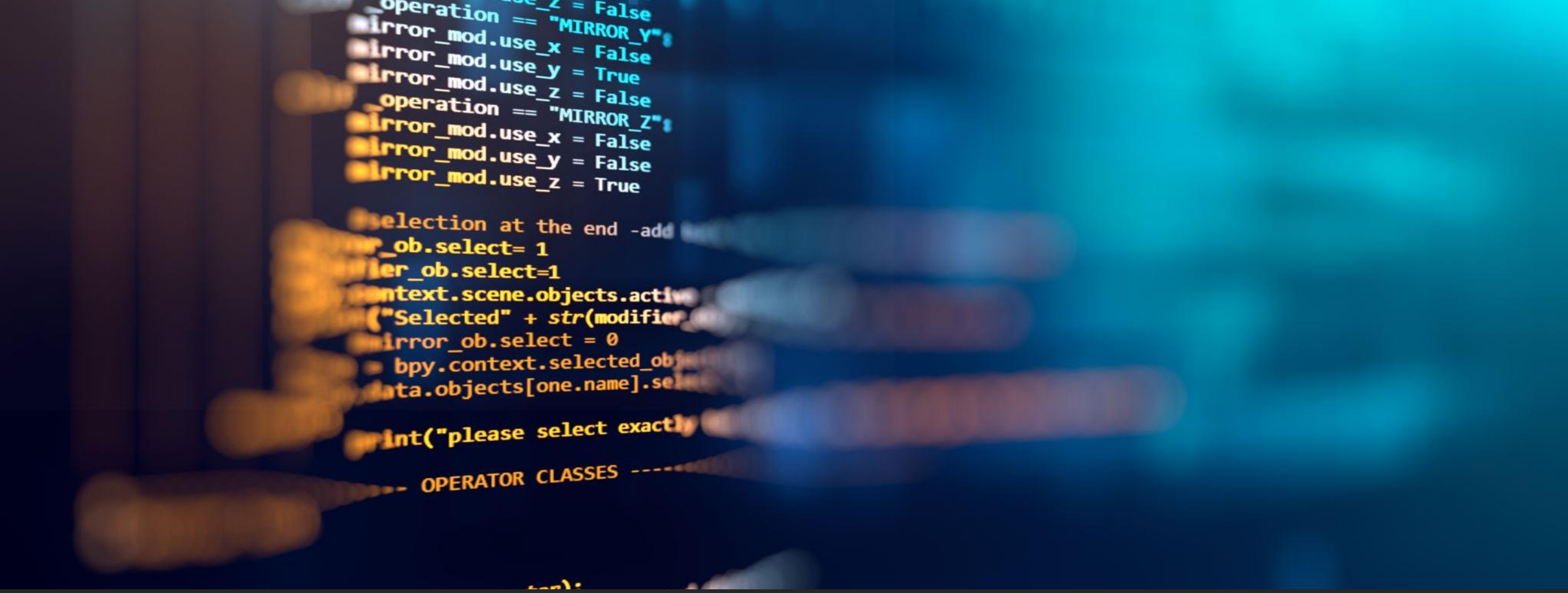
# Patrones de diseño – Command

**Problema:** Framework GUI

---

Estamos implementando un **framework** con los controles para ser utilizados en nuestra aplicación, hemos decidido empezar por la creación de un control de **Boton** que podrá ser utilizado para diferentes propósitos en nuestra aplicacion

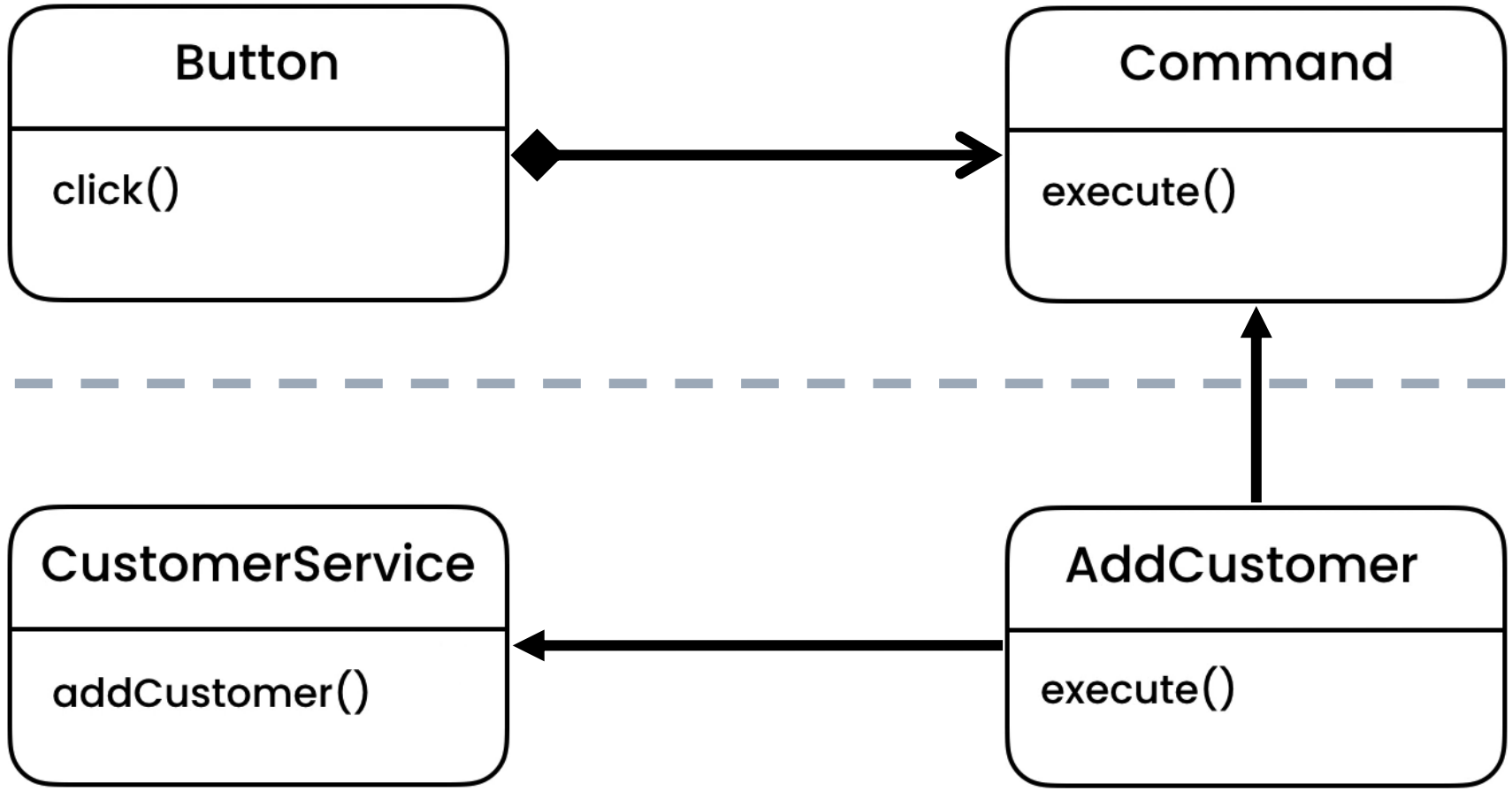





# Demo

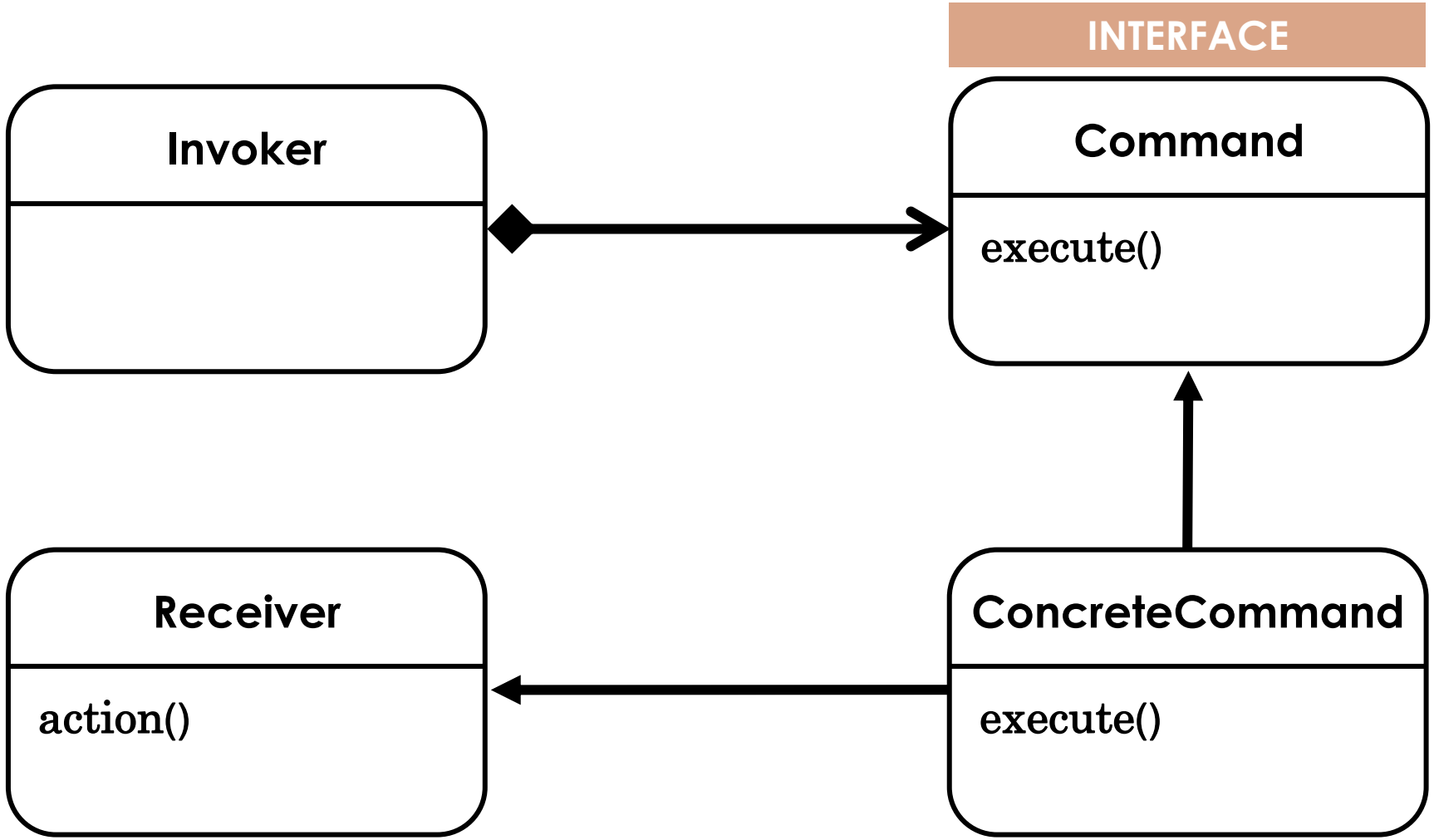
Revisemos en Código la implementación del problema

FRAMEWORK



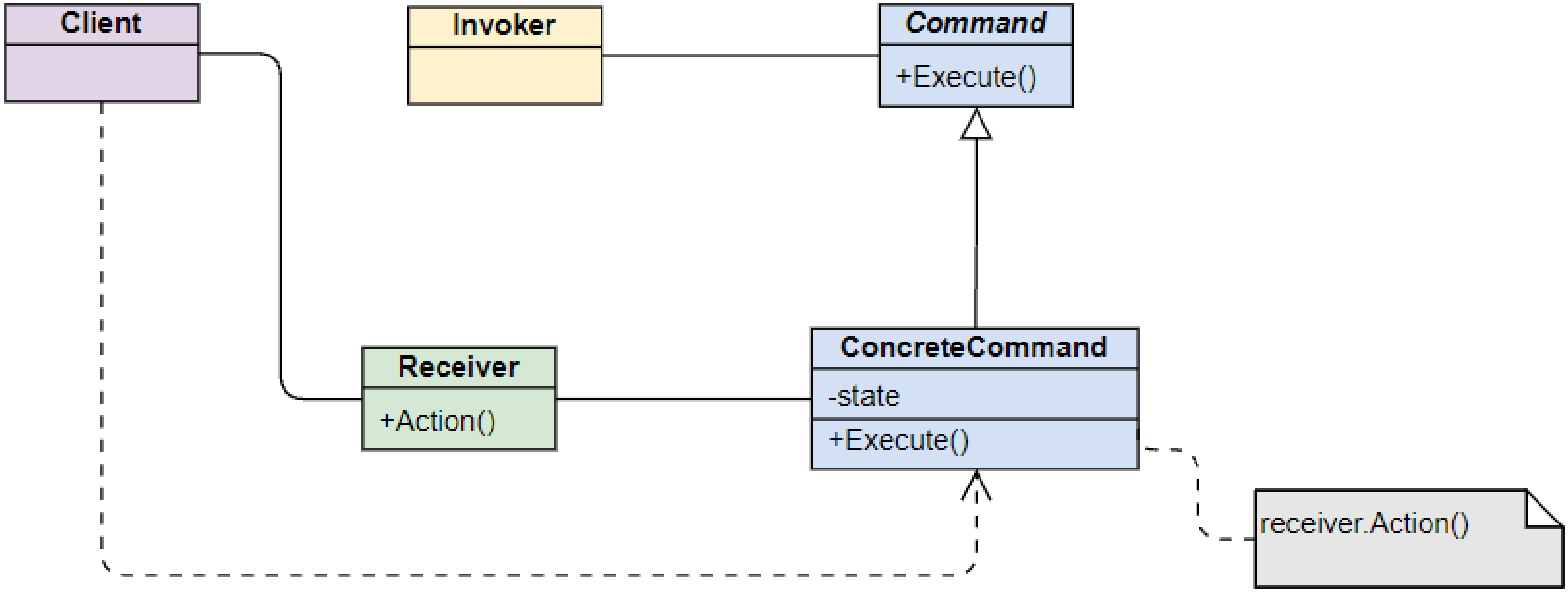

# Patrones de diseño – Command

Solución:




# Patrones de diseño – Command

Solución:

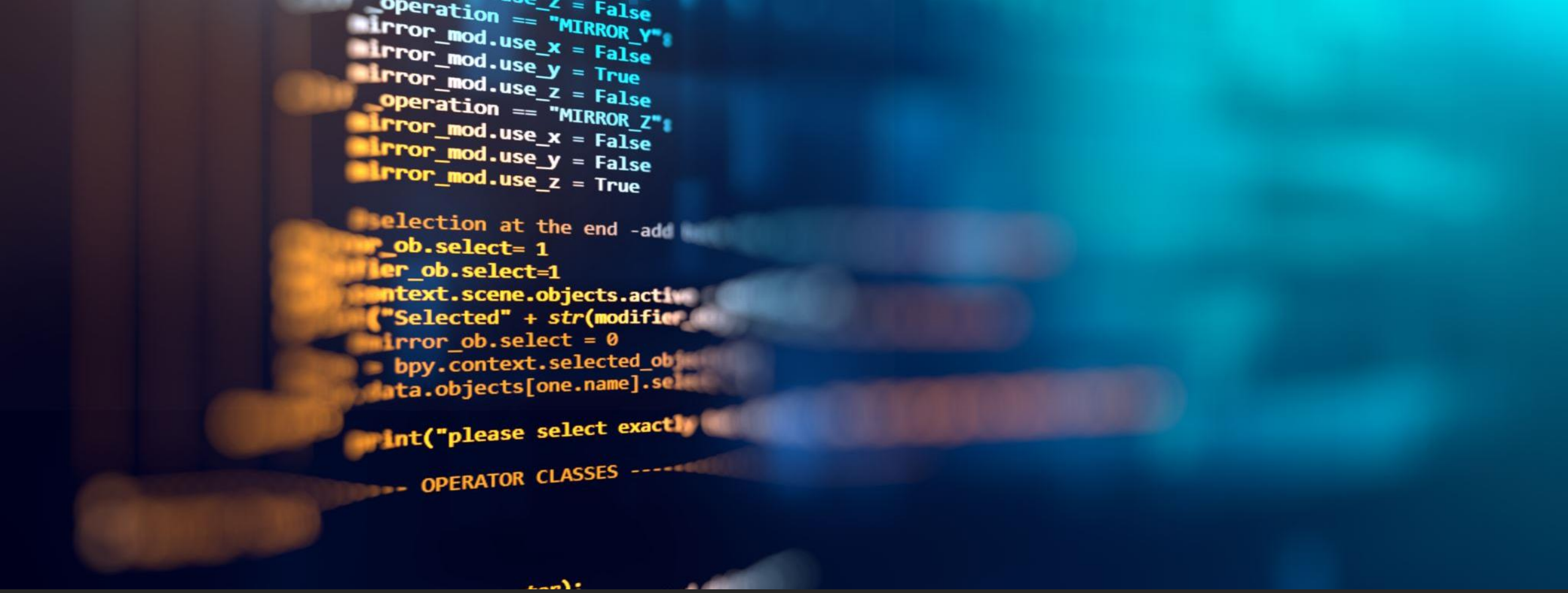



# Patron de diseño - Command

Estructura

# Patrones de diseño – Command

## Implementación




# Demo

Revisemos en Código la implementación de la solución

# Patrón de diseño Command

## Aplicabilidad

Cuándo utilizar este patrón?

Utiliza el patrón Command cuando quieras parametrizar objetos con operaciones.

Utiliza el patrón Command cuando quieras poner operaciones en cola, programar su ejecución, o ejecutarlas de forma remota.

Utiliza el patrón Command cuando quieras implementar operaciones reversibles.

# Patrones de diseño – Command

## VENTAJAS:

- ✓ **Principio de responsabilidad única**. Puedes desacoplar las clases que invocan operaciones de las que realizan esas operaciones.
- ✓ **Principio de abierto/cerrado**. Puedes introducir nuevos comandos en la aplicación sin descomponer el código cliente existente.
- ✓ Puedes implementar deshacer/rehacer.
- ✓ Puedes implementar la ejecución diferida de operaciones.
- ✓ Puedes ensamblar un grupo de comandos simples para crear uno complejo.

## DESVENTAJAS:

- × El código puede complicarse, ya que estás introduciendo una nueva capa entre emisores y receptores..

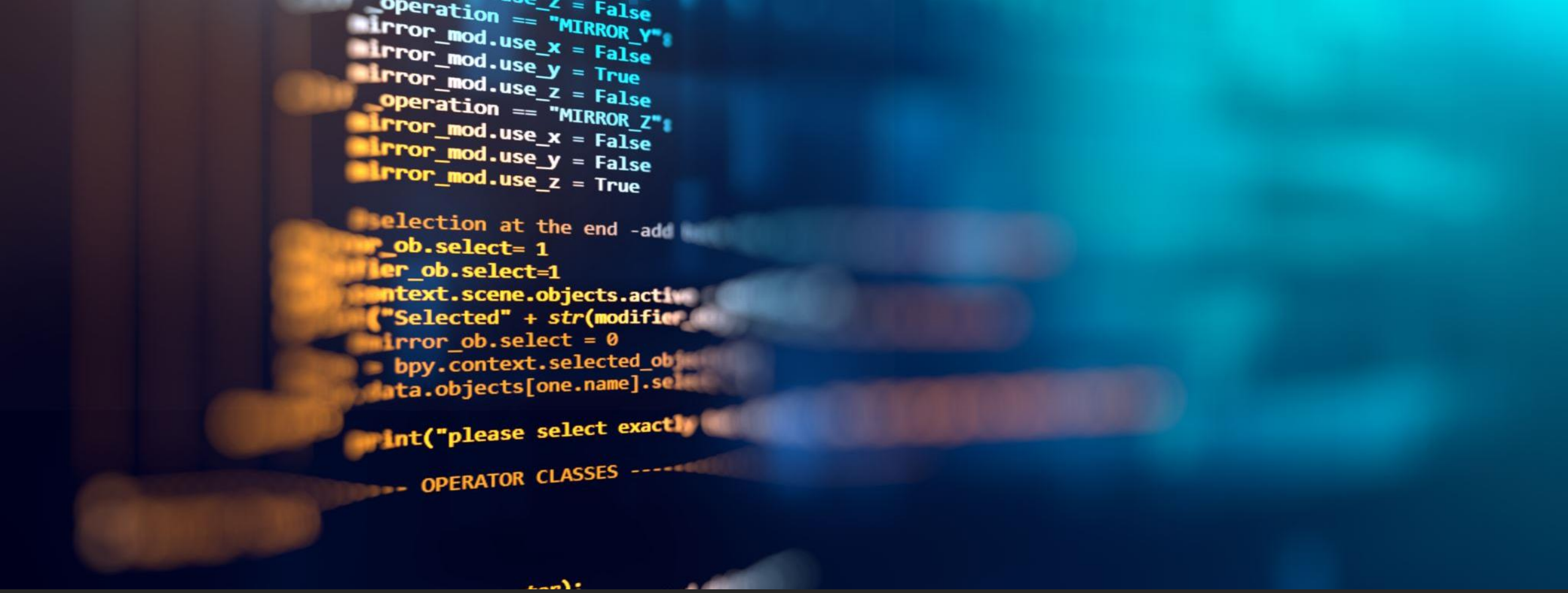
# Patrones de diseño – Command

## **Práctica:** Ejemplo Cajero

Imaginemos que tenemos un cajero que tiene 2 operaciones, una para Retiro y otra para Deposito.

Implementar utilizando el patrón Command, suponiendo que las 2 operaciones son comandos que actúan sobre el objeto cuenta, el cual debe tener los atributos NumeroCuenta y Saldo

Entonces cargaremos con un saldo inicial y operaremos sobre esa cuenta realizando las diferentes operaciones




# Demo

Solución de la tarea

# Patrones de diseño – Command

## Tarea: Ejemplo de un control remoto

Simular la implementación de un control Remoto, el cual tendrá 5 botones:

Implementar haciendo uso del Patrón Command teniendo en cuenta que se debe poder asociar a al menos 2 dispositivos:

1.- TV: Encender, Subir / Bajar Volumen, Subir / Bajar Canal
2.- DVD: Encender, Reproducir / Pausar, Adelantar / Retroceder

# Muchas Gracias por su atencion, los espero la siguiente clase