

Modificación Patrón Observer

Objetivo

Crear un nuevo notificador para cuando se modifique el nuevo atributo “Formato Moneda”

Resolución

A continuación, se detallarán los pasos realizados para completar el objetivo de la tarea

- i. En la clase “ConfigurationManager” definir un nuevo parámetro “currencyFormat”

```
public class ConfigurationManager extends AbstractObservable {
    private SimpleDateFormat defaultDateFormat;
    private NumberFormat moneyFormat;
    private String currencyFormat;
    private static ConfigurationManager configurationManager;

    private ConfigurationManager() { }

    public static ConfigurationManager getInstance() {
        if (configurationManager == null) {
            configurationManager = new ConfigurationManager();
        }
        return configurationManager;
    }

    public SimpleDateFormat getDefaultDateFormat() { return defaultDateFormat; }
    public void setDefaultDateFormat(SimpleDateFormat defaultDateFormat) {
        this.defaultDateFormat = defaultDateFormat;
        notifyAllObservers("defaultDateFormat", this);
    }

    public NumberFormat getMoneyFormat() { return moneyFormat; }
    public void setMoneyFormat(NumberFormat moneyFormat) {
        this.moneyFormat = moneyFormat;
        notifyAllObservers("moneyFormat", this);
    }

    public String getCurrencyFormat() { return currencyFormat; }
    public void setCurrencyFormat(String currencyFormat) {
        this.currencyFormat = currencyFormat;
        notifyAllObservers("currencyFormat", this);
    }
}
```

- ii. Creación de la nueva clase, que tendrá la funcionalidad para observar o notificar los cambios realizados sobre la propiedad para el formato de la moneda

```
public class CurrencyFormatObserver implements IObservable {
    @Override
    public void notifyObserver(String command, Object source) {
        if(command.equals("currencyFormat")){
            ConfigurationManager conf = (ConfigurationManager)source;
            System.out.println("Observer ==> "
                + CurrencyFormatObserver.currencyFormatChange > "
                + conf.getCurrencyFormat());
        }
    }
}
```

- iii. Sobre el método Main, se realizara las configuraciones del nuevo parámetro, así como la inicialización del nuevo observador; para revisar los resultados solicitados en la practica

```
public class ObserverMain {  
    public static void main(String[] args) {  
        ConfigurationManager conf = ConfigurationManager.getInstance();  
  
        //Se establecen los valores por default.  
        conf.setDefaultDateFormat(new SimpleDateFormat("yyyy/MM/dd"));  
        conf.setMoneyFormat(new DecimalFormat("##.00"));  
        conf.setCurrencyFormat("USD");  
        System.out.println("Established configuration");  
  
        //Se dan de alta lo observers  
        DateFormatObserver dateFormatObserver = new DateFormatObserver();  
        MoneyFormatObserver moneyFormatObserver = new MoneyFormatObserver();  
        CurrencyFormatObserver currencyFormatObserver = new  
CurrencyFormatObserver();  
        conf.addObserver(dateFormatObserver);  
        conf.addObserver(moneyFormatObserver);  
        conf.addObserver(currencyFormatObserver);  
        System.out.println("");  
  
        //Se cambia la fonfiguración  
        conf.setDefaultDateFormat(new SimpleDateFormat("dd/MM/yyyy"));  
        conf.setMoneyFormat(new DecimalFormat("###,##0.00"));  
        conf.setCurrencyFormat("EUR");  
        System.out.println("");  
  
        //Se realiza otro cambio en la configuración.  
        conf.setDefaultDateFormat(new SimpleDateFormat("MM/yyyy/dd"));  
        conf.setMoneyFormat(new DecimalFormat("###,##0"));  
        conf.setCurrencyFormat("USD");  
  
        conf.removeObserver(dateFormatObserver);  
        conf.removeObserver(moneyFormatObserver);  
        conf.removeObserver(currencyFormatObserver);  
        System.out.println("");  
  
        //Se realiza otro cambio en la configuración.  
        conf.setDefaultDateFormat(new SimpleDateFormat("MM/yyyy"));  
        conf.setMoneyFormat(new DecimalFormat("###,##0.00"));  
        conf.setCurrencyFormat("EUR");  
    }  
}
```

Patrón Observer

Por medio de la clase [CurrencyFormatObserver](#) que implementa la interfaz [IObserver](#), vamos a controlar los cambios que se realicen sobre el nuevo parámetro que se agrego a la clase [ConfigurationManager](#)

Una vez que se vincula el observador, seremos capaces de visualizar o notificar al usuario cada cambio realizado por medio del observador; La vinculación del observador se lo realiza por medio del método [addObserver\(\)](#)

Resultado

Finalmente se tiene el siguiente resultado con los cambios realizados

```
run:
Established configuration

Observer ==> DateFormatObserver.dateFormatChange > 25/07/2022
Observer ==> MoneyFormatObserver.moneyFormatChange > 01,11
Observer ==> CurrencyFormatObserver.currencyFormatChange > EUR

Observer ==> DateFormatObserver.dateFormatChange > 07/2022/25
Observer ==> MoneyFormatObserver.moneyFormatChange > 01
Observer ==> CurrencyFormatObserver.currencyFormatChange > USD

BUILD SUCCESSFUL (total time: 0 seconds)
```

