Open-Closed Principle

(You can add code but can't modify existing code.)

- Classes should be open for extension but closed for modification.

- When the business requirements change then the classes can be extended, but not modified.

- Interfaces are one way to follow OCP.

- A classes should be well written. So that it doesn’t have to be changed whenever the requirements change.

- If the OCP is applied well, then a program should be able to be changed by adding new code instead of changing existing code that already works.

Example

Payable.java

```
interface Payable
{

    public void pay();

}
```

CreditCard.java

```
class CreditCard implements Payable
    {

                @Override
                 public void pay()
                {
                    // Logic for credit card
                }

     }
```

PayPal.java

```
class PayPal implements Payable
    {
        @Override
         public void pay()
          {
              // Logic for Paypal
          }
     }
```

PaymentFactory.java

```
class PaymentFactory
{
    public void initPayment( Payable pi)
     {
        pi.pay ();
     }
}
```

Test.java

```
class Test
{
	public static void main(String args [])
	{
		PaymentFactory pf=null;

		Pay pay=new CreditCard();

		pf=new PaymentFactory();

		pf.initPayment (pay);

	}
}
```

In future if we want add another payment option then we can just implements Payable interface and create another class without modifying existing classes.

Code for extension

WirePay.java

```
class WirePay implements Pay
{

    @Override
    public void pay()
    {
        // logic for WirePay
    }

}
```