Monday, October 14, 2024 11:40 AM

Abstraction

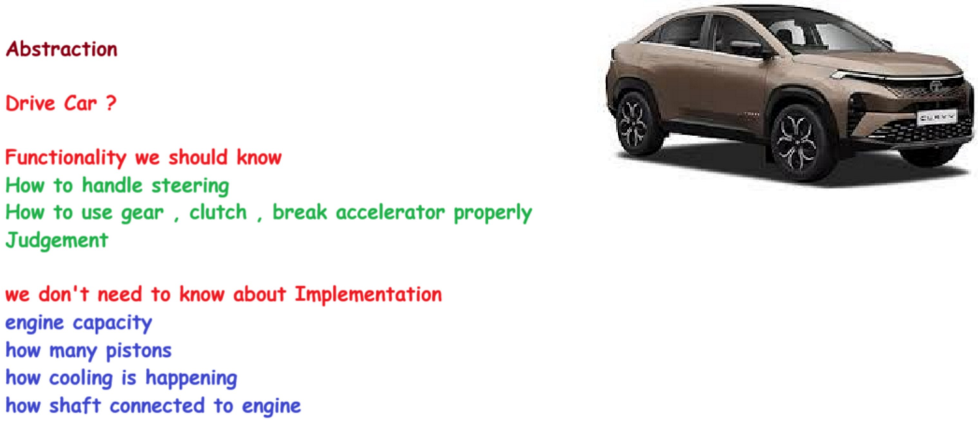
what is abstraction ?

-> abstraction is one of major pillar of oops concept

-> abstraction is a process of hiding implementation details and showing only the functionality to the user

-> to achieve abstraction in java interface and abstract class is use.

-> abstraction is use to create API's (Application Programming Interface) , project layouts and project standards .



Abstraction

Drive Car ?

Functionality we should know

How to handle steering

How to use gear , clutch , break accelerator properly

Judgement

we don't need to know about Implementation

engine capacity

how many pistons

how cooling is happening

how shaft connected to engine

What is API ?

-> API is a set of rules , protocols , tools that allow different software

-> there are three types API

Inbuilt API

-> inbuilt API is use for private purpose created in some programming language and use in same programming language

-> example: collection framework , jdbc framework, executer framework

Open Specification API

-> this are the shared API across various programming language meant for some general purpose requirements

-> example : Spring framework , Servlet, Apache Server, ORM tools

Third API

-> third party API is shared for different application to fulfil their requirement

-> IRCT, Weather Report, Yatra, Google Pay

What is interface ?

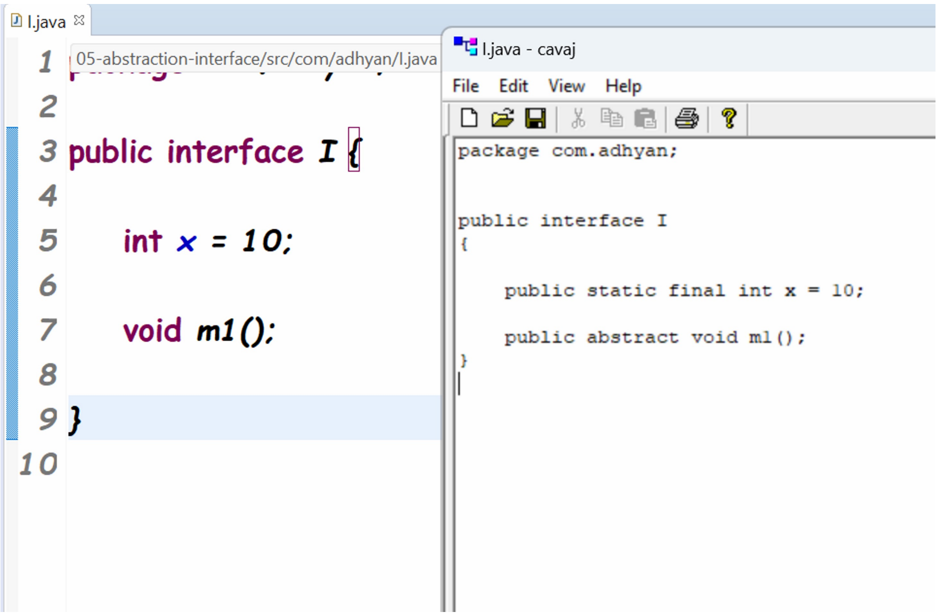
-> interface is set of rules and guidelines

-> all methods present in interface is by default public abstract

-> all variables declared in interface is by default public static final

-> interface dose not contains constructor

-> we can't create object of interface

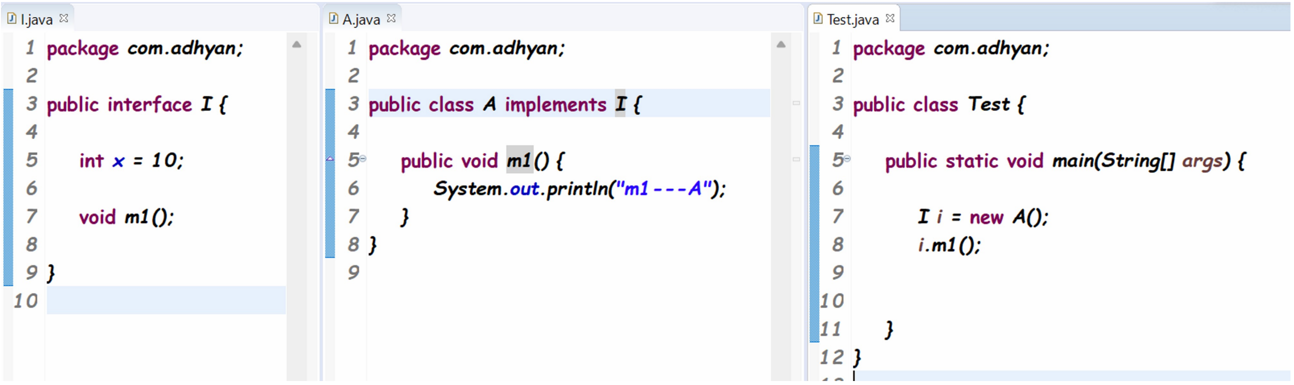


```
05-abstraction-interface/src/com/adhyan/I.java
```

```java

public interface I {

    int x = 10;

    void m1();

}
```

```java
package com.adhyan;


public interface I
{

    public static final int x = 10;

    public abstract void m1();
}
```

E.x



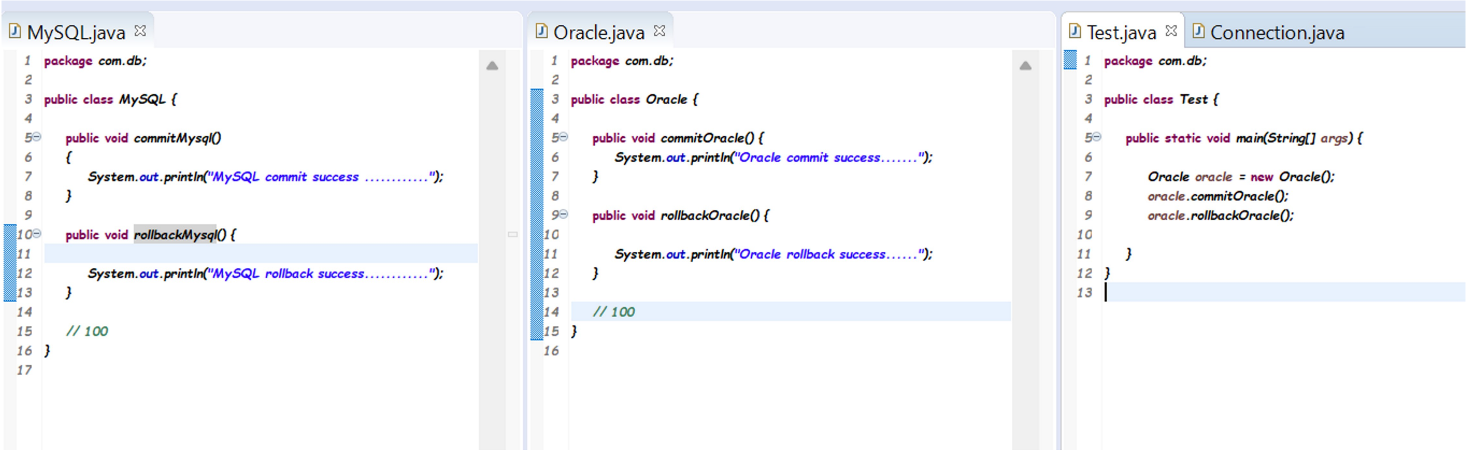
```java
package com.adhyan;

public interface I {

    int x = 10;

    void m1();

}
```

```java
package com.adhyan;

public class A implements I {

    public void m1() {
        System.out.println("m1 ---A");
    }
}
```

```java
package com.adhyan;

public class Test {

    public static void main(String[] args) {

        I i = new A();
        i.m1();


    }
}
```

Interface use case

1) Without interface

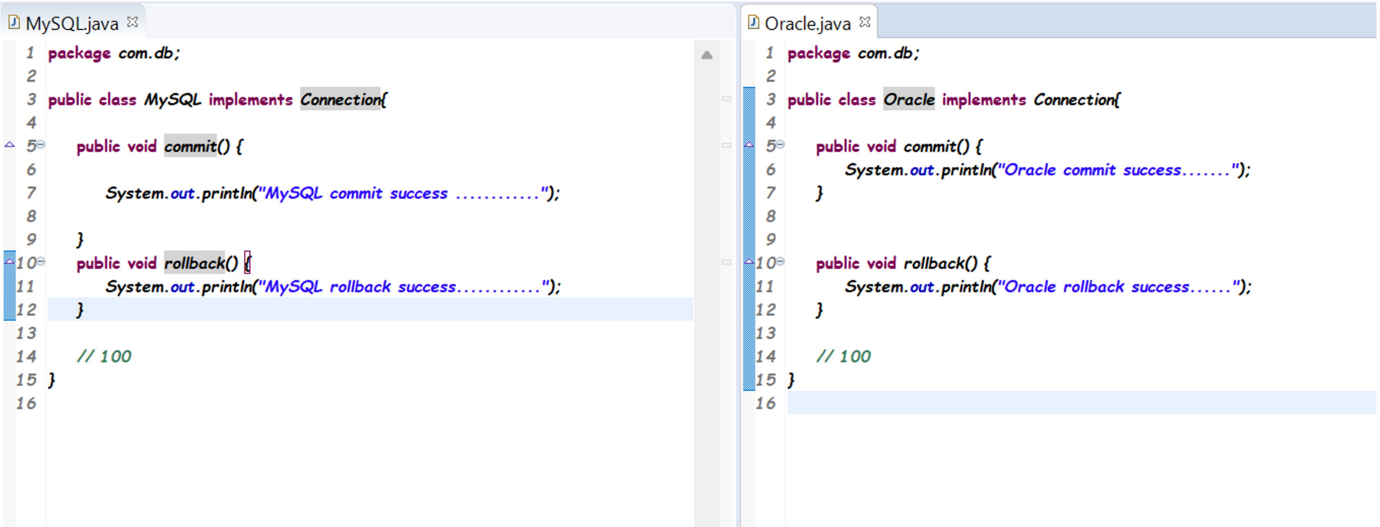


```java
package com.db;

public class MySQL {

    public void commitMysql()
    {
        System.out.println("MySQL commit success ...........");
    }

    public void rollbackMysql() {

        System.out.println("MySQL rollback success...........");
    }

    // 100
}
```

```java
package com.db;

public class Oracle {

    public void commitOracle() {
        System.out.println("Oracle commit success.......");
    }

    public void rollbackOracle() {

        System.out.println("Oracle rollback success......");
    }

    // 100
}
```

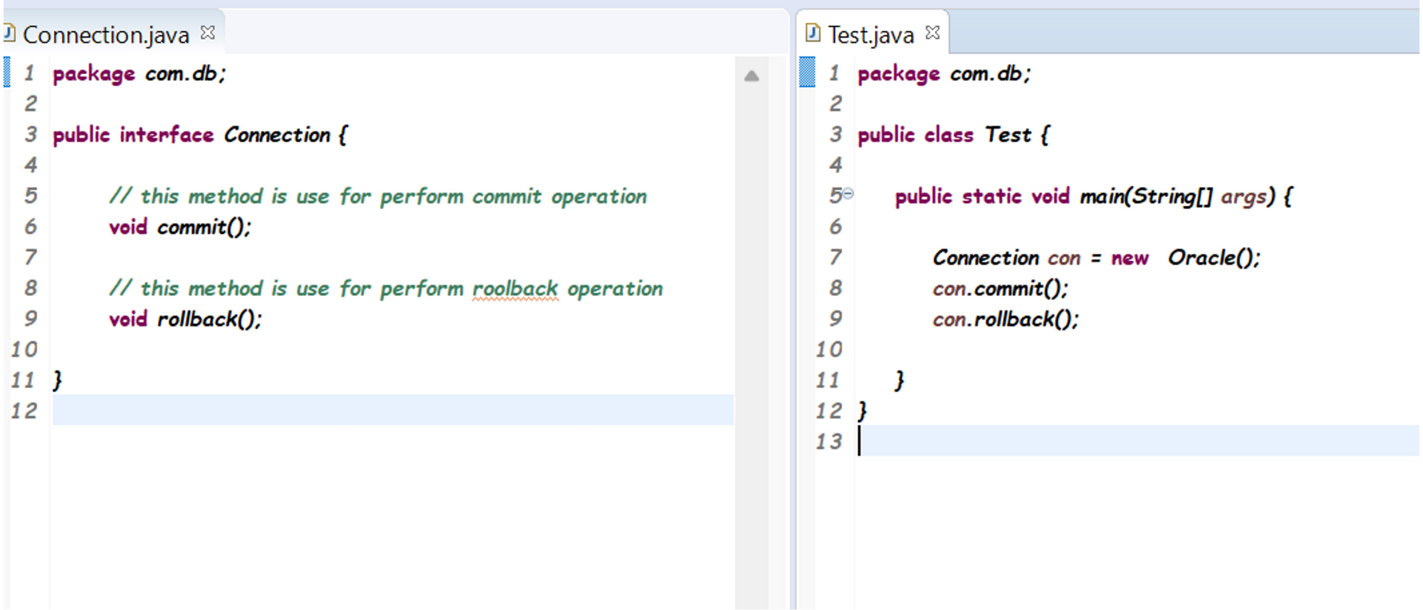
```java
package com.db;

public class Test {

    public static void main(String[] args) {

        Oracle oracle = new Oracle();
        oracle.commitOracle();
        oracle.rollbackOracle();

    }
}
```

2) With interface



```
package com.db;

public class MySQL implements Connection{

    public void commit() {

        System.out.println("MySQL commit success ............");

    }
    public void rollback() {
        System.out.println("MySQL rollback success............");
    }

    // 100
}
```

```
package com.db;

public class Oracle implements Connection{

    public void commit() {
        System.out.println("Oracle commit success.......");
    }


    public void rollback() {
        System.out.println("Oracle rollback success......");
    }

    // 100
}
```



```
package com.db;

public interface Connection {

    // this method is use for perform commit operation
    void commit();

    // this method is use for perform roolback operation
    void rollback();

}
```

```
package com.db;

public class Test {

    public static void main(String[] args) {

        Connection con = new  Oracle();
        con.commit();
        con.rollback();

    }
}
```