DAY-38

09 August 2023 16:01

WHY TO DECLARE METHODS AS ABSTRACT ?

- WHEN WE DON'T KNOW IMPLEMENTATION OF A METHOD AFTER THAT PARTICULAR TIME, SUCH METHODS CAN EDECLARED AS ABSTRACT.
- THE ADVANTAGE OF DECLARING A METHOD AS ABSTRACT IS SO THAT WE CAN PROVIDE MULTIPLE IMPLEMENTATION FOR THE METHOD IN ITS SUBCLASS.
- WE CAN NOT DECLARE STATIC METHOD AS ABSTRACT BECAUSE ABSTRACT IS APPLICABLE ONLY OBJECT LEVEL MEMBER FUNCTIONS, NOT CLASS LEVEL MEMBER FUNCTIONS.
- SINCE AN ABSTRACT CLASS CONTAIN BOTH CONCRETE AS WELL AS ABSTRACT METHODS, SUPPOSE IF THE USER WANTS TO ACCESS PROPERTIES OF ABSTRACT CLASS, THEN THERE ARE TWO POSSIBILITIES.
    1. IF THE ABSTRACT CLASS CONTAINS NON-STATIC PROPERTIES THEN USER HAS TO WAIT UNTIL SUBCLASS PROVIDES IMPLEMENTATION FOR THIS ABSTRACT CLASS AND LATER THEY CAN CREATE OBJECT OF SUBCLASS TO ACCESS THESE PROPERTIES.
    2. THAT IS WHY IT IS RECOMMENDED TO HAVE ALL THE CONCRETE PROPERTIES AS STATIC, SO THAT THEY CAN BE ACCESSED DIRECTLY USING CLASS NAME.
- WE CAN NOT DECLARE ABSTRACT CLASS AS FINAL BECAUSE CLASS WHICH ARE DECLARED AS FINAL WILL NOT HAVE ANY SUBCLASS, WHEREAS FOR ABSTRACT CLASS IT IS MANDATORY TO HAVE A SUBCLASS SO THAT THE IMPLEMENTATION CAN BE PROVIDED.
- WHILE PROVIDING IMPLEMENTATION FOR AN ABSTRACT METHOD OF ABSTRACT CLASS IN ITS SUBCLASS, WE SHOULD EITHER RETAUN SAME VISBILITY OF METHOD OR INCREASE IT, BUT IF WE TRY TO DECREASE VISBILITY, THEN COMPILER WILL THROW THE ERROR
- WE CAN NOT DECLARE ABSTRACT METHOD AS PRIVATE BECAUSE PRIVATE MEMBERS ARE NOT INHERITED TO ANOTHER CLASS WHEREAS IMPMENTATION OF ABSTRACT METHOD SHOULD ALWAYS BE IN SUBCLASS

```
package abstract1;

public abstract class Test6
{
	int x, y;

	public Test6(int x, int y)
	{
		this.x = x;
		this.y = y;
		System.out.println("inside the constrcutor of parent");
	}

	public abstract void add();
}


package abstract1;

public class Test7 extends Test6
{
	public Test7()
	{
		super(10,20);
		System.out.println("inside the constructor of child");
	}

	public void add()
	{
		System.out.println("the sum of 2 numbers is "+(1+1));
	}
	public static void main(String[] args)
	{

		Test7 a1 = new Test7();
		System.out.println(a1.x);
		System.out.println(a1.y);
		a1.add();
	}

}
```

- ABSTRACT CLASS WILL CONTAIN CONSTRUCTORS THOUGH WE CAN NOT CREATE OBJECT OF ABSTRACT CLASS, STILL TO COMPLETE CONSTRUCOTOR CHAINING PROCESS, THE ABSTRACT CLASS WILL HAVE CONSTRUCTORS.
- THE ABSTRACT CLASS CAN HAVE EITHER DEFAULT OR USER DEFINED CONSTRUCTORS.

INTERFACES

WHAT IS INTERFACE ?

- IT IS JAVA DEFENITION BLOCK WHICH CONTAINS DATA MEMBERS AND MEMBER FUNCTIONS.
- IN INTERFACE ALL THE DATA MEMBERS ARE BY DEFAULT **STATIC** AND **FINAL.**
- ALL THE MEMBER FUNCTIONS ARE BY DEFAULT **PUBLIC** AND **ABSTRACT**

```
	Interface interface_name
	{

		DataMembers;	(static and final)
		MemberFucntions; ( public and abstract)

	}
```

```
package interfaceDemo;

public interface Demo1
{
	int a = 1; //even if we do not specify , by default these will be static and final

	void add();//even if we do not specify , by default these will be public and abstract
	public static void main(String[] args)
	{
		System.out.println(a);
	}
}
```