Relationships in java

Here, we are going to connect one java class with another.

In one program, we can have N number of classes written but all of them cannot be executed at once.

There is a feature in java that we can connect multiple objects together, to form one bond.

For ex:

Object	*Properties*	*Behaviors*	*Relationship*
Car	Color, size, speed, company…	Drive	
Tire	Grip, pattern, speed ….	Spin for driving	Car has a tyre
Driver	Name, id, department	driveTheCar	Driver is driving the car

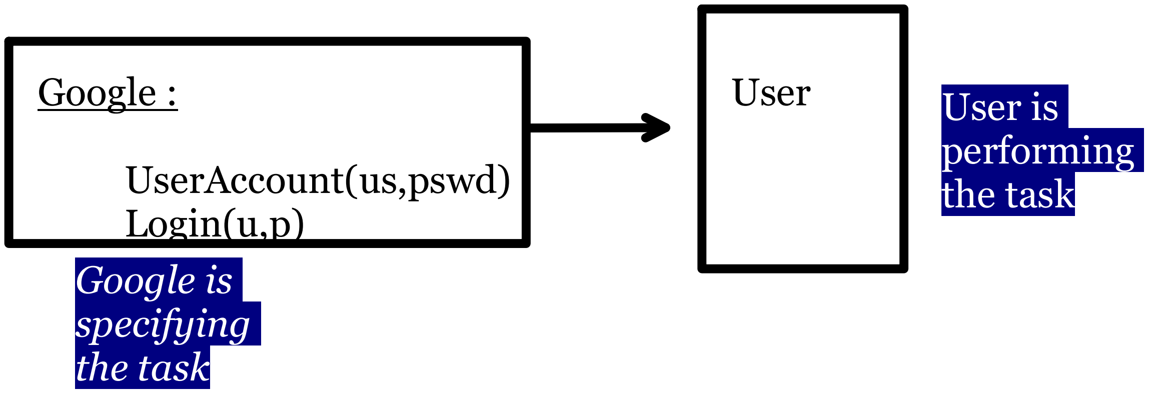
Composition / Aggregation/ Association:

Has a Relationship:

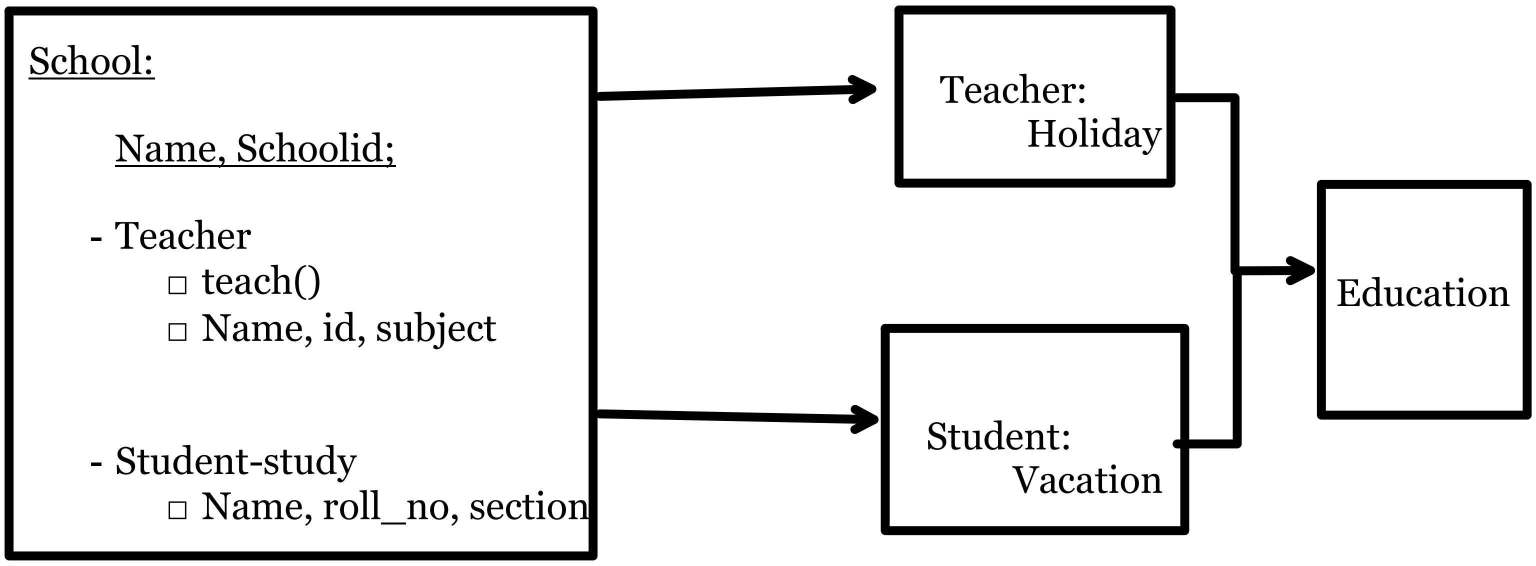
In this part of relationship, a whole is dependent on a part.

Here, there will be multiple classes which contains, their own features, the classes are written in such a way that, one will be the large object, another is going to be a part of it.

Whole		*Part*
Car	Has a	Tyre
Car	Has a	Driver
School	Has a	Student
School	Has a	Teacher
Google	Has a	User Account

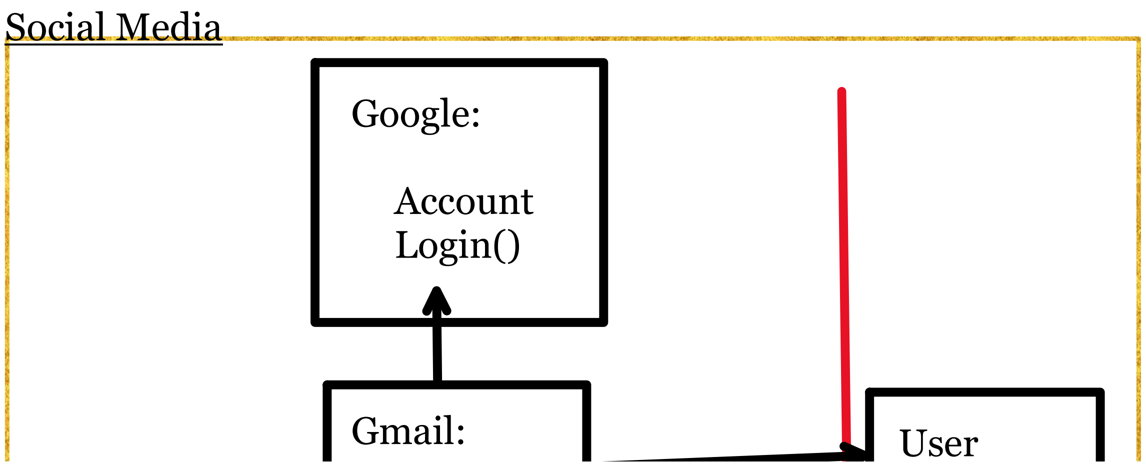


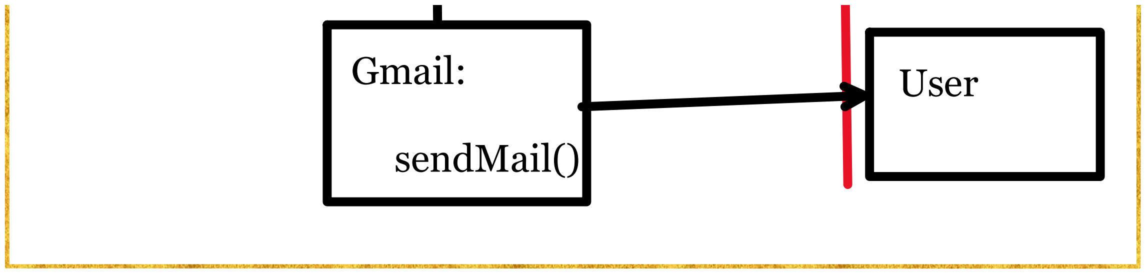
```
class Google//Driven class (specify the operation)
{
   String name, email, password;

   public Google(){}

   public Google(String name, String email, String password)
   {
      this.name=name;
      this.password=password;
      this.email=email;
   }

   {
      System.out.println("Account is created");
   }

   void Use()
   {
      System.out.println(name+" is using google");
   }

   public boolean login(String u, String p)
   {
      if( u==email && p==password)
      {
         System.out.println("Login Sucessfull");
         return true;
      }
      else
      {
         System.out.println("Invalid credentials");
         return false;
      }
   }
}

class User //Driver class (Perform the operation)
{
   public static void main(String[] args)
   {
      Google Acc=new Google("ABcd", "a@gmail.com", "qwerty");
      if(Acc.login("a@gmail.com", "qwerty"))
      {
         Acc.Use();
      }
   }
}
```



```
class School//driven class
{
   static String name="Prestige School", S_id="Abs001";
   String Tname, Sname, subject;
   int Tid, roll_no;
   char section;


   public School(){}
   //teacher
   public School(String Tname,int Tid, String subject)
   {
      this.Tid=Tid;
      this.Tname=Tname;
      this.subject=subject;
      System.out.println("Mr. "+Tname+" has joined "+name);
   }
   //student
   public School(String Sname, int roll_no, char section)
   {
      this.Sname=Sname;
      this.roll_no=roll_no;
      this.section=section;
      System.out.println(Sname+" is studying in "+name);
   }
   public void teach()
   {
      System.out.println(Tname+" is teaching "+subject);
   }
   public void study()
   {
      System.out.println(roll_no+", "+Sname+" is studying in "+section+" section");
   }
}
class teacher//driver
{
   String name;
   int id;
   School sc;
   public teacher() {}
   //the teacher is joining the school
   public teacher(String name, int id,String subject)
   {
      this(name, id);
      this.sc=new School(name, id, subject);
   }
   //a teacher is created
   public teacher(String name, int id)
   {
      this.name=name;
      this.id=id;
   }
   public void holiday()
```

```
    {
        System.out.println(name+" is on leave for today");
    }
}
class Student//driver
{
    String name;
    int roll_no;
    School sc;
    public Student() {}
    //student is getting admission in school
    public Student(String name, int roll_no,char section)
    {
        this(name, roll_no);
        this.sc=new School(name, roll_no,section);
    }

    //a student is getting created
    public Student(String name, int roll_no)
    {
        this.name=name;
        this.roll_no=roll_no;
    }

    public void vacation()
    {
        System.out.println("Vacations have started for "+name);
    }
}


class Education//driver to teacher/student
{
    static
    {
        System.out.println("Lets learn");
    }
    public static void main(String[] args)
    {
        teacher t=new teacher("Abc", 101,"Maths");
        //here teacher is created, along with school assigned with teacher
        Student s=new Student("Xyz", 1,'A');
        //here student is created, along with school assigned to student

        System.out.println("---------------");
        t.sc.teach();
        System.out.println("---------------");
        s.sc.study();
        System.out.println("---------------");
        t.holiday();
        System.out.println("---------------");
        s.vacation();
        System.out.println("---------------");
    }
}
```





In the above example, the user is the driver class.

But there are multiple driven classes, in which, google is larger and Gmail is part of google, but the part here is dependent on the whole.

The smaller object is dependent on a larger object.

This kind of relation where part is specified by whole is called as Is a relationship.
Basically here the part is going to be extension of whole

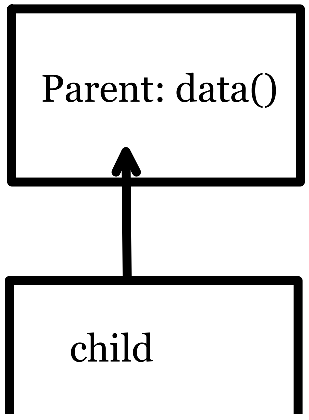
Ex:

Part		***Whole***
Gmail	***Is a***	Sub brand of Google
Car	***Is a***	Type of vehicle
Child	***Is***	Dependent on parent
Dog	***Is a***	Kind of animal
Chennai	***Is a***	City of Tamil Nadu

If we want to implement the *is a relationship* then we have to perform inheritance.

Inheritance:

Inheritance is a process of a child class accessing the data of parent class.



The inheritance or is a relationship in java can be implemented using the ***extends*** keyword.

```
Class A                 ------->parent class
{
   ///data///
}
```

```
   ///data///
}

Class B extends A ----> child class
{
   ///use the data of A///
}
```

- When we use extends keyword, the classes are connected, so in execution, both parent and child are going to get class area memories.
- There is a keyword called ***super.*** And a super call statement. It creates a reference/instance of parent

```
Class Parent
{
        //data//

        Constructor(){}

}

Class Child extends Parent
{
        //data//

        Constructor()
        {
                //super();
        }
}
```

this		Current class
super		Parent class
this()		Constructor of current class
super()		Constructor of parent class

Instanceof : just checks whether the instance is present or not.

```
class parent
{
   public parent()
   {
      System.out.println("Instance created for parent");
   }
}
class child extends parent
{
   public child()
   {
      //super();

      System.out.println("Instance created for child");
   }
}
class driver
{
   public static void main(String[] args)
   {
      child c=new child();

      System.out.println( c instanceof child);//true
      System.out.println( c instanceof parent);//true
      //System.out.println( c instanceof driver);//false
   }
}
```

```
class Parent
{
   String name ="Dad";
   void work()
   {
      System.out.println(name+" : Parent working for family");
   }
}

class child extends Parent
{
   String name="son";

   void enjoy()
   {
      System.out.println(name+" : Child is enjoying because of parent");
   }
   public static void main(String[] args)
   {
      child c= new child();

      c.work();
      c.enjoy();
   }
}
```