

# Job Sheet 04 - Relation Class

## I. Competence

After to go through main discussion This, student capable:

1. Understand draft relation class;
2. Implementing association relationship to in program.

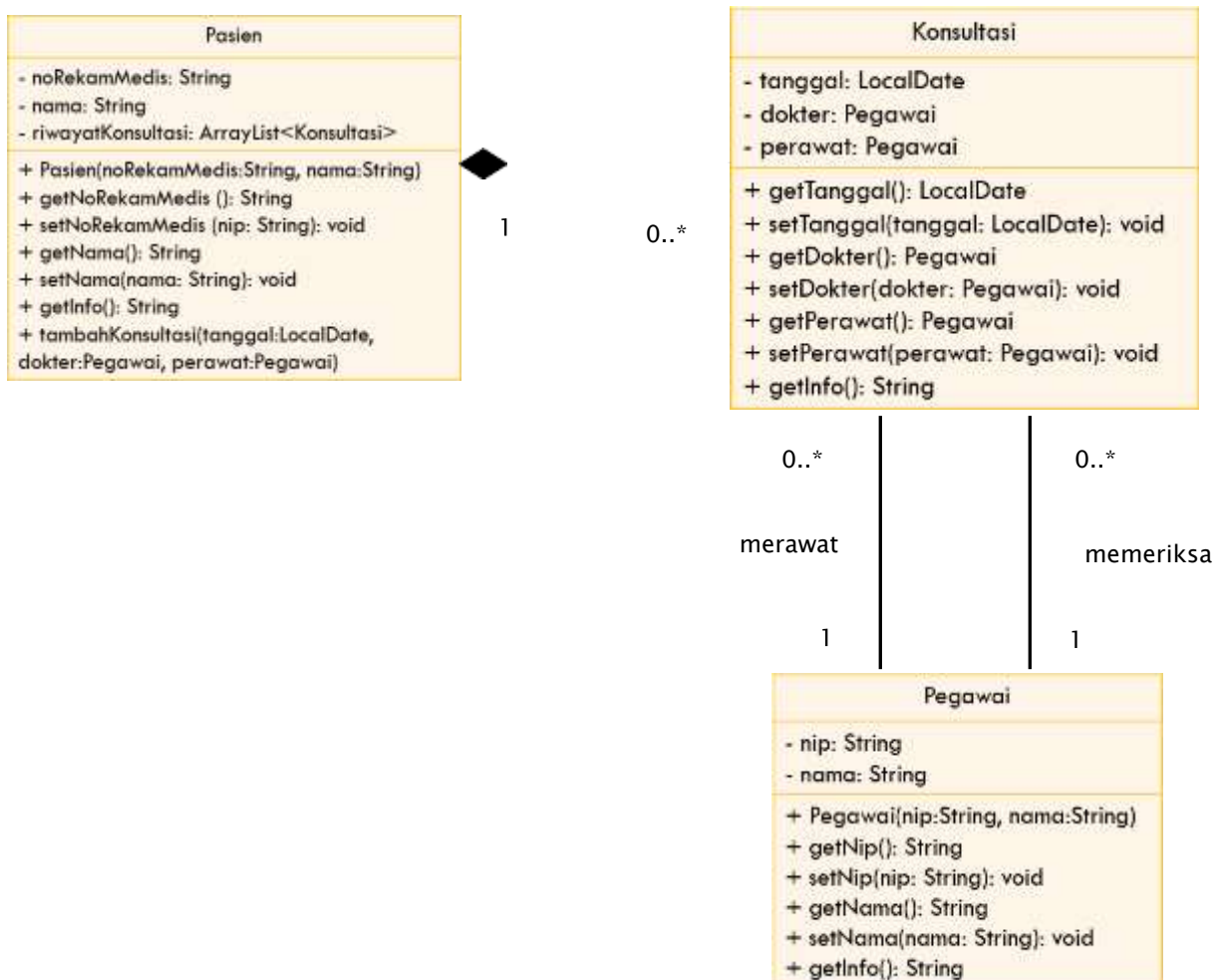
## II. Introduction

In more complex cases, more than one will be found in a system. *class* Which each other own relatedness between *class* One with Which other. On previous experiments, the majority of cases that have been worked on only focus on One *class* just. On job sheet This will done test Which involving a number of *class* Which each other related.

## III. Practicum

On practical work This will developed a system information House Sick Whichsave data history patient consultation.

Take note *class* diagram following:



- a. Make it folder new with Name Hospital
- b. Make it class Employee. Add attribute nip And Name on class Employee with access modifier private

```
public class Pegawai {  
    private String nip;  
    private String nama;  
}
```

- c. Make it *constructor* For class Employee with nip parameters And Name.

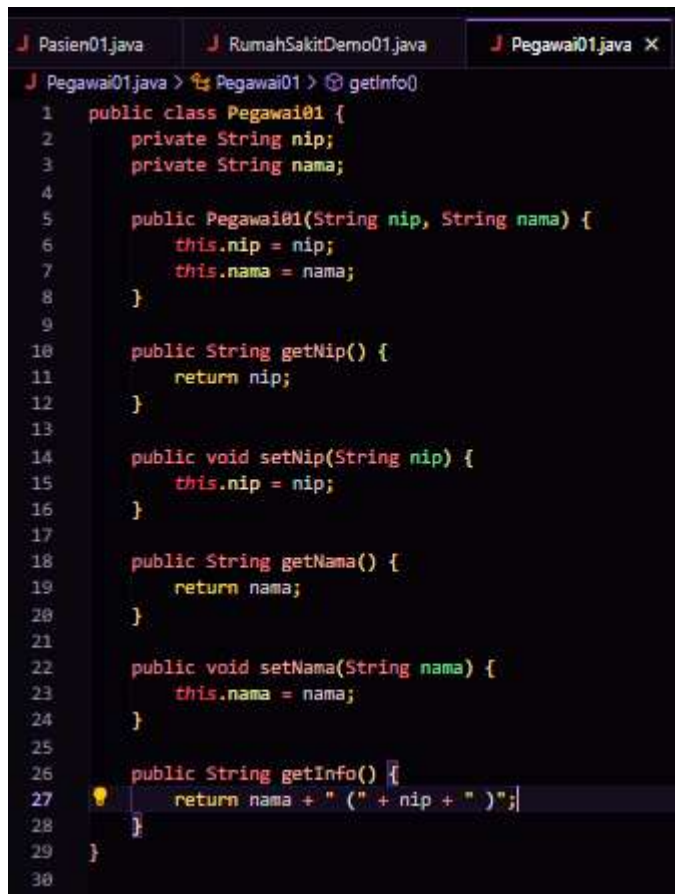
```
public Pegawai(String nip, String nama) {  
    this.nip = nip;  
    this.nama = nama;  
}
```

- d. Implement **setter** And **getter** For class Employee.

```
public String getNip() {  
    return nip;  
}  
  
public void setNip(String nip) {  
    this.nip = nip;  
}  
  
public String getNama() {  
    return nama;  
}  
  
public void setNama(String nama) {  
    this.nama = nama;  
}
```

- e. Implement *method* getInfo() as following:

```
public String getInfo() {  
    return nama + " (" + nip + ")";  
}
```



```
Pasien01.java  RumahSakitDemo01.java  Pegawai01.java X
Pegawai01.java > Pegawai01 > getInfo()
1  public class Pegawai01 {
2      private String nip;
3      private String nama;
4
5      public Pegawai01(String nip, String nama) {
6          this.nip = nip;
7          this.nama = nama;
8      }
9
10     public String getNip() {
11         return nip;
12     }
13
14     public void setNip(String nip) {
15         this.nip = nip;
16     }
17
18     public String getNama() {
19         return nama;
20     }
21
22     public void setNama(String nama) {
23         this.nama = nama;
24     }
25
26     public String getInfo() {
27         return nama + " (" + nip + ")";
28     }
29 }
30
```

- f. Furthermore make it class Patient Then add attribute noMedicalRecord Andname in the Patient class with the access level modifier private. Also provide setters and getter for second attribute the.

```

public class Pasien {
    private String noRekamMedis;
    private String nama;

    public String getNoRekamMedis() {
        return noRekamMedis;
    }

    public void setNoRekamMedis(String noRekamMedis) {
        this.noRekamMedis = noRekamMedis;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }
}

```

- g. Make it constructor For class Patient with noRekorMedic parameter And Name

```

public Pasien(String noRekamMedis, String nama) {
    this.noRekamMedis = noRekamMedis;
    this.nama = nama;
}

```

- h. Implement *method* getInfo()as following:

```

public String getInfo() {
    String info = "";
    info += "No Rekam Medis      : " + this.noRekamMedis + "\n";
    info += "Nama                  : " + this.nama + "\n";
    info += "\n";

    return info;
}

```

- i. This system will store data on every consultation that the patient has. Patients can conduct consultations more than once. Therefore, the consultation data will be stored in ArrayList form from objects Which type Consultation.
- j. Make it class with Name Consultation with attribute date type LocalDate, doctor is of type Employee, and nurse is of type Employee. Set access level modifier private For all over attribute. Do import java.time.LocalDate so that can to declare attribute date type LocalDate.

```
import java.time.LocalDate;

public class Konsultasi {
    private LocalDate tanggal;
    private Pegawai dokter;
    private Pegawai perawat;
}
```

- k. Provide setter And getter For each attribute on class Consultation

```
public LocalDate getTanggal() {
    return tanggal;
}

public void setTanggal(LocalDate tanggal) {
    this.tanggal = tanggal;
}

public Pegawai getDokter() {
    return dokter;
}

public void setDokter(Pegawai dokter) {
    this.dokter = dokter;
}

public Pegawai getPerawat() {
    return perawat;
}

public void setPerawat(Pegawai perawat) {
    this.perawat = perawat;
}
```

- l. Implement method getInfo() as following:

```
public String getInfo() {
    String info = "";
    info += "\tTanggal: " + tanggal;
    info += ", Dokter: " + dokter.getInfo();
    info += ", Perawat: " + perawat.getInfo();
    info += "\n";

    return info;
}
```

```

Konsultasi01.java > ...
1  import java.time.LocalDate;
2
3  public class Konsultasi01 {
4      private LocalDate tanggal;
5      private Pegawai01 dokter;
6      private Pegawai01 perawat;
7
8      public LocalDate getTanggal() {
9          return tanggal;
10     }
11
12     public void setTanggal(LocalDate tanggal) {
13         this.tanggal = tanggal;
14     }
15
16     public Pegawai01 getDokter() {
17         return dokter;
18     }
19
20     public void setDokter(Pegawai01 dokter) {
21         this.dokter = dokter;
22     }
23
24     public Pegawai01 getPerawat() {
25         return perawat;
26     }
27
28     public void setPerawat(Pegawai01 perawat) {
29         this.perawat = perawat;
30     }
31
32     public String getInfo() {
33         return String.format(format: "\tTanggal: %s, Dokter: %s, Perawat: %s\n",
34                                tanggal,
35                                dokter.getInfo(),
36                                perawat.getInfo());
37     }
38 }

```

- m. For keep data history consultation patient, so add attribute historyConsultation on class Patient with type arrayList<Consultation>. Attribute This will store a series of objects of type Consultation. Import java.util.ArrayList so that can to declare attribute type ArrayList of object.

```
private String noRekamMedis;
private String nama;
private ArrayList<Konsultasi> riwayatKonsultasi;
```

- n. Make it constructor parameterized For class Patient. Initiation mark attribute noRekamMedis and name based on name attribute. Instantiate/create new ArrayList For attribute historyConsultation;

```
public Pasien(String noRekamMedis, String nama) {
    this.noRekamMedis = noRekamMedis;
    this.nama = nama;
    this.riwayatKonsultasi = new ArrayList<Konsultasi>();
}
```

- o. Do import java.time.LocalDate so that can to declare attribute date type LocalDate on class Patient. Furthermore, implement method addConsultation() as follows:

```
public void tambahKonsultasi(LocalDate tanggal, Pegawai dokter, Pegawai perawat){
    Konsultasi konsultasi = new Konsultasi();
    konsultasi.setTanggal(tanggal);
    konsultasi.setDokter(dokter);
    konsultasi.setPerawat(perawat);
    riwayatKonsultasi.add(konsultasi);
}
```

- p. Modification method getInfo() For return info patient And list consultationever done

```
public String getInfo(){
    String info = "";
    info += "No Rekam Medis      : " + this.noRekamMedis + "\n";
    info += "Nama                  : " + this.nama + "\n";

    if (!riwayatKonsultasi.isEmpty()) {
        info += "Riwayat Konsultasi :\n";

        for (Konsultasi konsultasi : riwayatKonsultasi) {
            info += konsultasi.getInfo();
        }
    }
    else{
        info += "Belum ada riwayat konsultasi";
    }

    info += "\n";

    return info;
}
```



```

1  Pasien01.java  RumahSakitDemo01.java  Pegawai01.java  Konsultasi01.java
2  Pasien01.java > Pasien01 > Pasien01(String, String)
3
4  public class Pasien01 {
5      private String noRekamMedis;
6      private String nama;
7      private ArrayList<Konsultasi01> riwayatKonsultasi;
8
9      public Pasien01(String noRekamMedis, String nama) {
10         this.noRekamMedis = noRekamMedis;
11         this.nama = nama;
12         this.riwayatKonsultasi = new ArrayList<>(); // Inisialisasi daftar riwayat konsultasi
13     }
14
15     public String getNoRekamMedis() {
16         return noRekamMedis;
17     }
18
19     public void setNoRekamMedis(String noRekamMedis) {
20         this.noRekamMedis = noRekamMedis;
21     }
22
23     public String getNama() {
24         return nama;
25     }
26
27     public void setNama(String nama) {
28         this.nama = nama;
29     }
30
31     public ArrayList<Konsultasi01> getRiwayatKonsultasi() {
32         return riwayatKonsultasi;
33     }
34
35     public void setRiwayatKonsultasi(ArrayList<Konsultasi01> riwayatKonsultasi) {
36         this.riwayatKonsultasi = riwayatKonsultasi;
37     }
38
39     // Menambahkan konsultasi baru ke daftar riwayat
40     public void tambahKonsultasi(LocalDate tanggal, Pegawai01 dokter, Pegawai01 perawat) {
41         Konsultasi01 konsultasi = new Konsultasi01();
42         konsultasi.setTanggal(tanggal);
43         konsultasi.setDokter(dokter);
44         konsultasi.setPerawat(perawat);
45         riwayatKonsultasi.add(konsultasi);
46     }
47
48     // Mendapatkan informasi lengkap tentang pasien dan riwayat konsultasi
49     public String getInfo() {
50         StringBuilder info = new StringBuilder();
51         info.append(str:"No Rekam Medis: ").append(this.noRekamMedis).append(str:"\n");
52         info.append(str:"Nama: ").append(this.nama).append(str:"\n");
53
54         if (riwayatKonsultasi.isEmpty()) {
55             info.append(str:"Riwayat Konsultasi:\n");
56             for (Konsultasi01 konsultasi : riwayatKonsultasi) {
57                 info.append(konsultasi.getInfo());
58             }
59         } else {
60             info.append(str:"Belum ada riwayat konsultasi\n");
61         }
62     }
63 }

```

- q. Do import java.time.LocalDate so that can to declare attribute date type LocalDate on class HospitalDemo. Test program Which Already made with make objects on class HospitalDemo. Instantiation object new type Employee with Name ani use constructor Employee(String nip, String Name) with mark argument nip “1234” And Name “dr. Ani”. Continue instantiation object as follows:



```

public class RumahSakitDemo {
    Run | Debug
    public static void main(String[] args) {
        Pegawai ani = new Pegawai("1234", "dr. Ani");
        Pegawai bagus = new Pegawai("4567", "dr. Bagus");

        Pegawai desi = new Pegawai("1234", "Ns. Desi");
        Pegawai eka = new Pegawai("4567", "Ns. Eka");

        Pasien pasien1 = new Pasien("343298", "Puspa Widya");
        pasien1.tambahKonsultasi(LocalDate.of(2021, 8, 11), ani, desi);
        pasien1.tambahKonsultasi(LocalDate.of(2021, 9, 11), bagus, eka);

        System.out.println(pasien1.getInfo());

        Pasien pasien2 = new Pasien("997744", "Yenny Anggraeni");
        System.out.println(pasien2.getInfo());
    }
}

```

```

J Pasien01.java  J RumahSakitDemo01.java  J Pegawai01.java  J Konsultasi01.java
J RumahSakitDemo01.java > ts RumahSakitDemo01 > main(String[])
1  import java.time.LocalDate;
2
3  public class RumahSakitDemo01 {
4      Run | Debug
      public static void main(String[] args) {
5          Pegawai01 ani = new Pegawai01(nip:"1234", nama:"dr. Ani");
6          Pegawai01 bagus = new Pegawai01(nip:"4567", nama:"dr. Bagus");
7          Pegawai01 desi = new Pegawai01(nip:"7890", nama:"Ns. Desi");
8          Pegawai01 eka = new Pegawai01(nip:"6789", nama:"Ns. Eka");
9
10         Pasien01 pasien1 = new Pasien01(noRekamMedis:"343298", nama:"Puspa Widya");
11
12         // Menambahkan konsultasi untuk pasien
13         pasien1.tambahKonsultasi(LocalDate.of(year:2021, month:8, dayOfMonth:11), ani, desi);
14         pasien1.tambahKonsultasi(LocalDate.of(year:2021, month:9, dayOfMonth:11), bagus, eka);
15
16         // Menampilkan informasi pasien
17         System.out.println(pasien1.getInfo());
18
19         // Instansiasi pasien kedua
20         Pasien01 pasien2 = new Pasien01(noRekamMedis:"997744", nama:"Yenny Anggraeni");
21
22         // Menampilkan informasi pasien kedua
23         System.out.println(pasien2.getInfo());
24     }
25 }
26

```

- r. *Compile* Then *run* HospitalDemo and obtained results like following:

```
No Rekam Medis      : Puspa Widya
Nama                : 343298
Riwayat Konsultasi :
    Tanggal: 2021-08-11, Dokter: dr. Ani (1234), Perawat: Ns. Desi (1234)
    Tanggal: 2021-09-11, Dokter: dr. Bagus (4567), Perawat: Ns. Eka (4567)

No Rekam Medis      : Yenny Anggraeni
Nama                : 997744
Belum ada riwayat konsultasi
```

```
ahSakitDemo01'
No Rekam Medis: 343298
Nama: Puspa Widya
Riwayat Konsultasi:
    Tanggal: 2021-08-11, Dokter: dr. Ani (1234 ), Perawat: Ns. Desi (7898 )
    Tanggal: 2021-09-11, Dokter: dr. Bagus (4567 ), Perawat: Ns. Eka (6789 )

No Rekam Medis: 997744
Nama: Yenny Anggraeni
Belum ada riwayat konsultasi
```

## Question

Based on test 1, answer it questions Which related:

1. In in *class* Employee, Patient, And Consultation, there is method *setter* And *getter* For each its attributes. Whether the use of *method setter* And *getter* the ?

**The setter and getter methods have the following uses:**

- **Getter:** Used to retrieve values from private attributes of an object. This allows access to data without changing the attribute value directly.
  - **Setter:** Used to change the value of an object's private attributes. This allows for modification of the data while providing the opportunity to add additional validation or logic when the data is changed.
2. In in *class* Consultation No in a way explicit there is constructor withparameter. Is this means class Consultation No own constructor?  
**No, the Consulting class still has a default parameterless constructor, which is automatically provided by Java if we don't explicitly declare the constructor.**
  3. Take note *class* Consultation, attribute where just Which type *object* ?  
**A attribute of type object :**
    - **doctor**
    - **nurse**
    -
  4. Take note *class* Consultation, on line which one Which show that *class* Consultation own relation with *class* Employee?
    - **private Doctor's staff; — This attribute indicates that the Consultation have a relationship with the Employee as a doctor.**
    - **private Employee nurse; — This attribute indicates that the Consultation also has a relationship with the Employee as a nurse.**

5. Take note on *class* Patient, What Which done by code `consultation.getInfo()` ?

**In the Patient class , the `consult.getInfo()` code is used to:**

- **Retrieving Consultation Details:** Call the `getInfo()` method to get information about the consultation.
- **Displaying Information:** The results are added to the patient information for display .

6. On method `getInfo()` in class Patient, there is line code:

`if (!consultationhistory.isEmpty())`

Whether Which done by line the?

**The code line `if (!rihiwatKonsultasi.isEmpty())` checks whether the `sejarahKonsultasi` list has data.**

- **If Data Exists:** The code inside the if block will be executed to display the consultation history information.
- **If Empty:** The code inside the else block will be executed to state that there is no consultation history.

7. In the Patient class constructor, there is a line of code:

`this.historyConsultation = new ArrayList<>();`

Whether Which done by line the? Whether Which happen If line theremoved?

**The code line `this.rihiwatKonsultasi = new ArrayList<>();` in the Patient class constructor :**

- **Initialize** the `historyConsultation` attribute with a new `ArrayList` object .

**If removed:**

- `historyConsultation` will be null , causing an error when trying to add or access the data.

#### IV. Task

Implement studies the case that has made on PBO duties Theory to in program