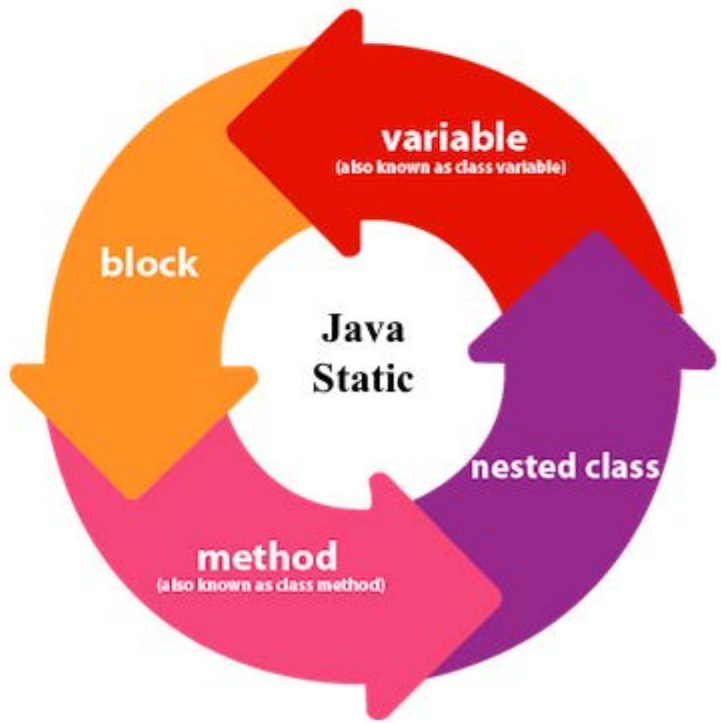
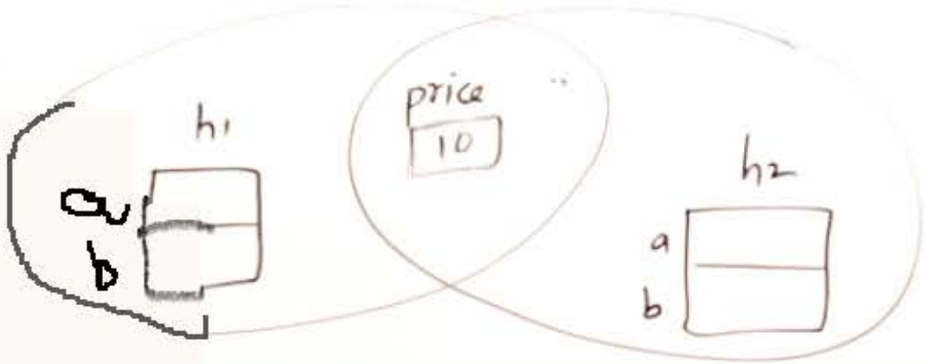
Static

--> Static Keyword is used for representing Meta Data (data about data).
--> It is useful for representing the information of a class.
--> Static members belongs to a class and they can be shared by all the objects of the class and all the objects have their own non-static members .

- Static Variables
- Static Methods
- Static Nested class
- Static Blocks



Static



--> Static variable can be used to refer to the common property of all objects (which is not unique for each object), for example, the company name of employees, college name of students, etc.

--> Static variable related to class not to only object.

--> Static variable & method created in method area not in heap & stack.

--> static variable memory mai sirf ek baar create hota, uska baad ham uska kitni baar use kar sakta hai. But non-static variable ka jitna baar object banta hai utni baar different memory location allot hota hai.

--> static variable ko haam bina as a objected create kia bhi direct access kar sakta hai (jaisa ise case mai Hondacity.price).

```
class HondaCity
{
    static long price=10;
    int a,b;
}
class Test
{
    p.s.v.main()
    {
        HondaCity h1=new HondaCity();
        HondaCity h2=new HondaCity();
        S.o.p(h1.price); —— 10
        S.o.p(h2.price); —— 10
        h1.price=20;
        S.o.p(HondaCity.price);
    }
}
```

Static

```
class Test
{
  p.s.v.main()
  {
    s.o.p(HondaCity.onRoadPrice("delhi"));
    s.o.p(HondaCity.price);
  }
}
```

```
class HondaCity
{
  static long price=10;
  int a,b;

  static double onRoadPrice(String city)
  {
    switch(city)
    {
      case "delhi":
          return price+price*0.1;
      case "mumbai":
          return price+price*0.09;
    }
  }
}
```

--> Static method class ka sirf static member ko hi access.

--> Inside static mtd. this & super jisa keyword use nahi hota hai.

--> Ine mth. ko direct call kar sakta hai aur object bna ka bhi.

Static Block

```
class My
{
    static int s;

    static
    {
        s.o.p("Block1");
        s=10;
    }
    .
    .
    .
    static
    {
        s.o.p("Block2");
    }
    .
    .
    .
}
```

--> Set of statements are written in the form of blocks and are made static.

--> iska use kisi bhi static variable ko initialize karna ka liya hota hai jab hmna ek large piece of code likha ho.

--> Practically iska use bahut kam hi hota hai.

--> Jo bhi code static block mai hoga chahai wo kahi bhi define kyu nahi hai sab sai phela wahi call hoga. Uska baad hi main mtd. (or class) ka code run hoga.

final keyword

```
class My
{
    final int MIN=1;        (1)
    final int NORMAL;
    final int MAX;

(2) static
    {
        NORMAL=5;
    }

    My()
    {
(3)     MAX=10;
    } MIN=0;
    ...
}
```

--> Final keyword use karna ka baad, ham use variable ki value update nahi kar sakta hai.

--> Final keyword ka use karta hua jab koi variable name define karta hai to wo capital letter mai hona jaruri hai.

--> Ise 3 trah sai initialize kar sakta hai:

i) directly

ii) static block ka andhar

iii) kisi constructor ka andhar

final Keyword

Final Method

```
class Super
{
    final void meth1()
    {
        s.o.p("Hello");
    }
}

class Sub extends Super
{
    void meth1()
    {
        s.o.p("Hi");
    }

    void meth2()
    {
        s.o.p("Bye");
    }
}
```

--> Final method ko override nahi kar sakta hai.

--> Final class mai bhi na to kuch change kar sakta hai aur na to final class ko extend hi kar sakta hai.

final class

```
final class Super
{
    =
    =
    =
}

class Sub extends Super
{
    =
    =
    =
}
```

Keyword	Usage	Details
`static`	Class-level members	- **Variables**: Shared among all instances. - **Methods**: Can be called without an instance. - **Blocks**: Used for class initialization.
`final`	Constants and restrictions	- **Variables**: Immutable values. - **Methods**: Cannot be overridden. - **Classes**: Cannot be extended.