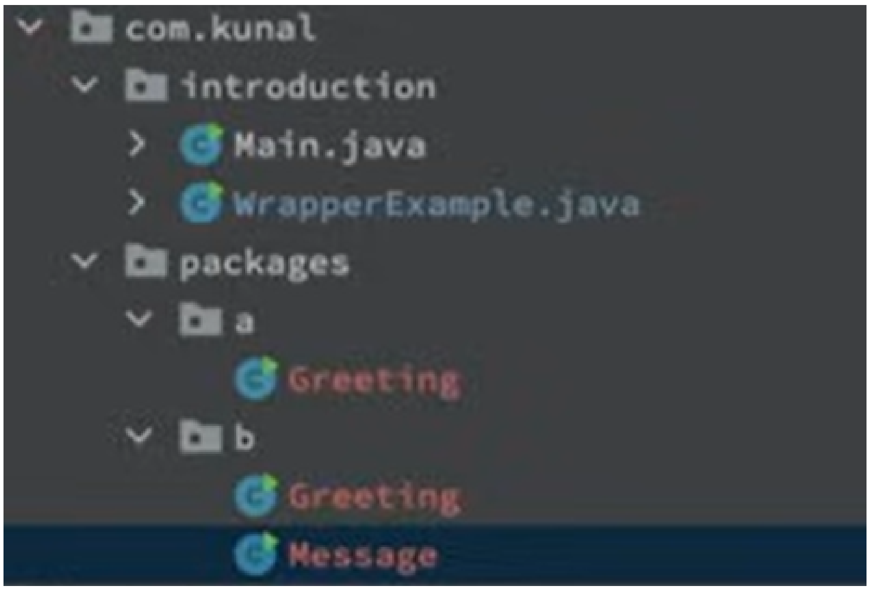
2️⃣

Packages, Static, Singleton Class, In-built Methods

Packages

- Stored in the Hierarchical method
- Containers for classes
- Used to have classes compartment-wise

Importing



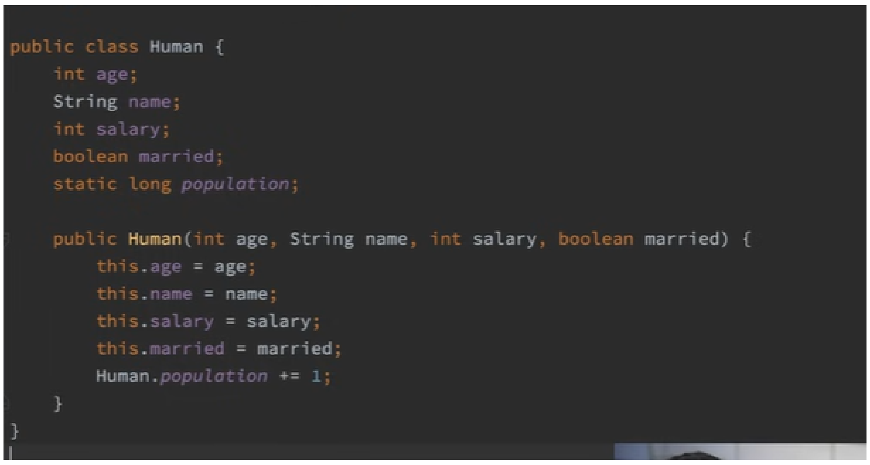
```
com.kunal
    introduction
        Main.java
        WrapperExample.java
    packages
        a
            Greeting
        b
            Greeting
            Message
```

If the method I need to use in a current class/file exists in the same folder, we don’t import anything.

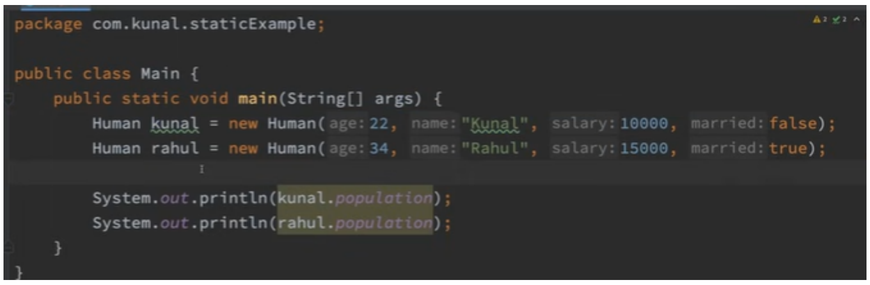
while if the class/file exists in another folder, WE HAVE TO IMPORT the class in the current folder specifying their path

Static Variable

- The world's population is 7 billion. This is a fact shared by all human beings on earth.
- If Earth's population is 7 Billion for an object 'Aish'. it remains the same even for all other objects.
- The property POPULATION is OBJECT INDEPENDENT. it doesn't depend on the object or change for each object.
- **These kinds of PROPERTIES are called Static variables / Static methods**
- Static variable: when a member is declared as static in a method. it can be accessed without even an object being created and also without referencing the object (Independent of objects)



```java
public class Human {
    int age;
    String name;
    int salary;
    boolean married;
    static long population;

    public Human(int age, String name, int salary, boolean married) {
        this.age = age;
        this.name = name;
        this.salary = salary;
        this.married = married;
        Human.population += 1;
    }
}
```



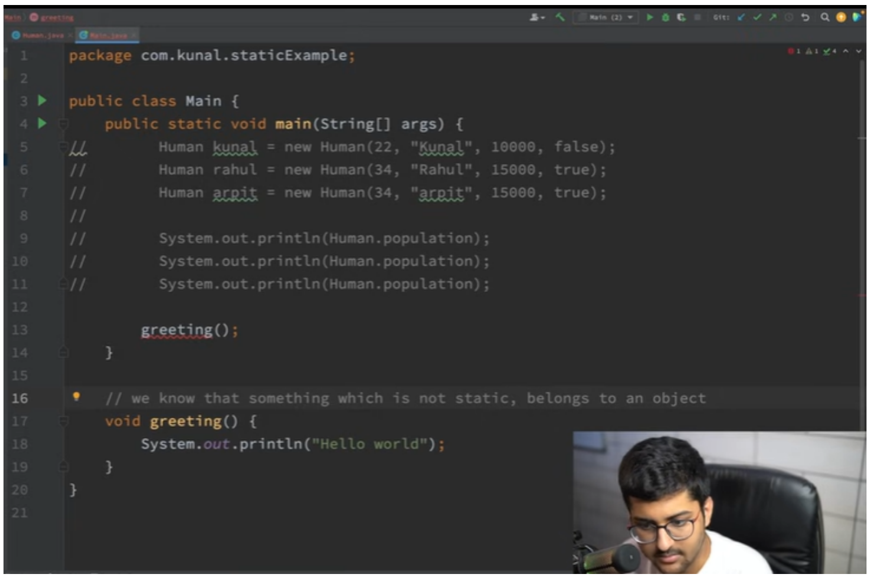
```java
package com.kunal.staticExample;

public class Main {
    public static void main(String[] args) {
        Human kunal = new Human( age: 22,  name: "Kunal",  salary: 10000,  married: false);
        Human rahul = new Human( age: 34,  name: "Rahul",  salary: 15000,  married: true);

        System.out.println(kunal.population);
        System.out.println(rahul.population);
    }
}
```

Static Method

Example: Main method (can be accessed without creating an object)

- **Suppose STATIC from the main method is removed. How can the program be run to access something in a main class without being an object created?**

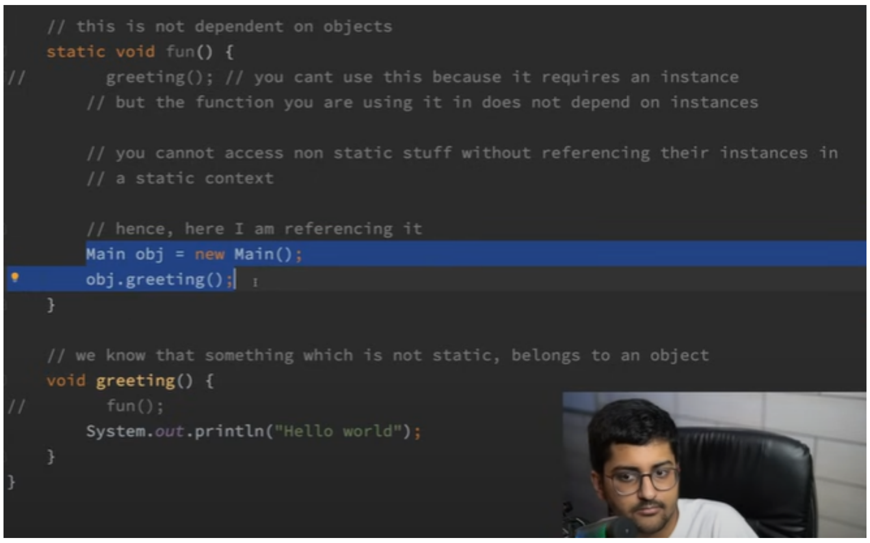
- If the main method isn't Static, we would be supposed to create an object mandatorily to access or run a program's MAIN METHOD in a class.



```java
package com.kunal.staticExample;

public class Main {
    public static void main(String[] args) {
//        Human kunal = new Human(22, "Kunal", 10000, false);
//        Human rahul = new Human(34, "Rahul", 15000, true);
//        Human arpit = new Human(34, "arpit", 15000, true);
//
//        System.out.println(Human.population);
//        System.out.println(Human.population);
//        System.out.println(Human.population);

        greeting();
    }

    // we know that something which is not static, belongs to an object
    void greeting() {
        System.out.println("Hello world");
    }
}
```

📌 **Non-static methods /Variables Cannot be used under or inside Static Methods/Variables. (Vice versa works out)**

Here, 'public static void main(String[] args)' is a static method that is independent of any object whereas the method 'greeting' is non-static and is dependent on an object.



```java
    // this is not dependent on objects
    static void fun() {
//        greeting(); // you cant use this because it requires an instance
        // but the function you are using it in does not depend on instances

        // you cannot access non static stuff without referencing their instances in
        // a static context

        // hence, here I am referencing it
        Main obj = new Main();
        obj.greeting();
    }

    // we know that something which is not static, belongs to an object
    void greeting() {
//        fun();
        System.out.println("Hello world");
    }
}
```

Initialising the Static Varibales

```java
public class staticBlock{
    static int a=10;
    static int b;
}

static{
    System.out.println("This is the static block");
    b=a*3;
}

public static void main(String[] args){
    staticBlock obj1=new staticBlock();
    System.out.println(staticBlock.a + "," + staticBlock.b);

    staticBlock.b +=5;
    System.out.println(staticBlock.a + "," + staticBlock.b);


    // another object
    staticBlock obj2=new staticBlock();
    System.out.println(staticBlock.a + "," + staticBlock.b);
}
```

This gives output as:

This is a static block

10, 30

10, 35

10, 35

> ***Here, the static method will run only once when the class is loaded with the object***

Consider,

```
public class innerClass{
    class trail{

    }
}
```

The outer class Cannot be STATIC, while the inner class can be either STATIC or no :

Singleton Classes

- class that can create only a single object
- prevent using constructor(when constructor is called , new obj is created)

```
public class Singleton(){
    private Singleton(){

    }
}
```