13-Static Method

What is a Static Method?

In Java, the static keyword is used to create methods that belong to the class rather than to instances of the class. This means that static methods can be called without creating an object of the class. Any method declared with the static keyword is referred to as a static method.

Features of Static Methods

- **Class-Level Method**: A static method is associated with the class itself, not with any particular object instance.
- **Access by All Instances**: Every instance of a class can access a static method.
- **No Need for Object Reference**: Static methods can access static variables directly, without needing to refer to an instance of the class.
- **Limited Access**: A static method can only access other static data (static variables and static methods). It cannot access instance variables directly.
- **Direct Access**: Static methods can be accessed directly within both static and non-static methods.

Syntax to Declare a Static Method

```
Access_modifier static void methodName() {
    // Method body
}
```

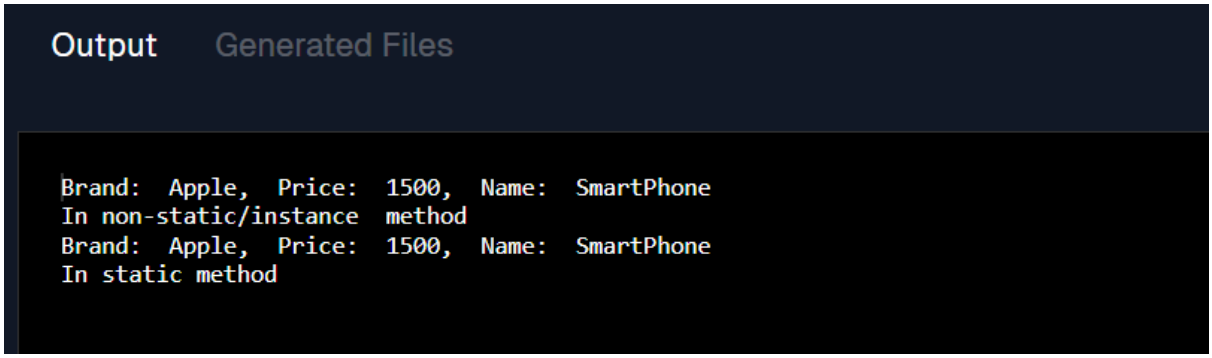
Example: Using Static Methods in a Class

```
 Class Mobile
ass Mobile {
  // Instance variable
  String brand;
  int price;
  static String name;

  // Instance method
  public void show() {
      // Prints the brand, price, and name of the mobile
      System.out.println("Brand:  " + brand + ",  Price:  " + price + ",  Name:  " + name);
        System.out.println("In non-static/instance  method");
  }

  // Static method
  public static void show1(Mobile obj) {
      // Prints the brand, price, and name of the mobile
      System.out.println("Brand:  " + obj.brand + ",  Price:  " + obj.price + ",  Name:  " + name);
      // Prints a message indicating that this is a static method
      System.out.println("In static method");
  }


 Class Demo
blic class Demo {
  public static void main(String[] args) {
      // Creates a new instance of the Mobile class
      Mobile obj = new Mobile();
      // Sets the brand and price of the mobile
      obj.brand = "Apple";
      obj.price = 1500;
      // Sets the name of the mobile
      Mobile.name = "SmartPhone";

      // Calls the instance method of the Mobile class
      obj.show();
      // Calls the static method of the Mobile class with the current instance as an argument
      Mobile.show1(obj);
  }
```

Explanation of the Code

1. **Instance Method (show()):**
    - The show() method is an instance method and is called using an object (obj) of the Mobile class.
    - It can directly access instance variables (brand, price) and static variables (name).
2. **Static Method (show1()):**
    - The show1() method is declared static. It can only directly access static variables (name).
    - To access instance variables (brand, price), we pass an object of the Mobile class reference as a parameter.
    - Inside show1(), we use the passed object (obj) to access the instance variables.
3. **Main Method:**

- The main() method is also static. It is the entry point of the program and does not require an object of the class to be called.
- The main() method creates an instance of Mobile, assigns values to its fields, and calls both the instance and static methods.



Why is the Main Method in Java Static?

The main() method is static because it is the entry point for the program. When the program starts, there is no object of the class available. If the main() method were non-static, the JVM would need to create an object of the class before it could call the method, which would create a circular dependency and deadlock like situation. Making the main() method static avoids this issue and allows the program to start without needing to create an object.

FAQs

1. **Can a static method access non-static variables?**
   - No, a static method cannot directly access non-static (instance) variables. It can only access them through an object reference.
2. **Why do we need to pass an object to a static method?**
   - We pass an object to a static method to access instance variables and instance methods. Static methods can only interact with static data directly.
3. **Can we override static methods?**
   - No, static methods cannot be overridden. However, they can be hidden by declaring a static method with the same name in a subclass.
4. **What happens if we remove the static keyword from the main() method?**
   - If you remove the static keyword from the main() method, the program will not run because the JVM will not have a valid entry point to start the program.