ifference between static and non-static nested class in Java

Now, let's ee some notable differences between static and non-static nested classes in Java:

1. Nested static class doesn't need reference of Outer class but a non-static nested class or Inner class requires Outer class reference. You can not create an instance of the Inner class without creating an instance of the Outer class. This is by far the most important thing to consider while making a nested class static or non-static. You can also see these free Java courses to learn more about essential Java concepts like static and non-static members, variables, and classes

2. static class is actually a static member of the class and can be used in the static context, like a static method or a static block of Outer class.

3. Another difference between static and non-static nested classes is that you can not access non-static members, like method and field, into the nested static class directly. If you do, you will get errors like "non-static member can not be used in the static context." While the Inner class can access both static and nonstatic members of the Outer class.

Here is the code sample of using both *nested static class vs. nonstatic class* :

```
/**
 * Java program to demonstrate What is nested static and non-
static class.
 * How to create instances of the static and non-static classes
and How to call
 * methods of nested static and Inner class in Java. Overall
comparison of
 * static vs. non-static class.
 */
class Outer{
    private static String message = "HelloWorld";

    // Static nested class
    private static class MessagePrinter{
        //Only static member of Outer class is directly accessible
in a nested static class

        public void printMessage(){
            // Compile time error if message field is not static
            System.out.println("Message from nested static class :
" + message);
        }
    }

    //non-static nested class - also called Inner class
    private class Inner{

        // Both static and non-static member of Outer class is
accessible in this Inner class
        public void display(){
```

```
            System.out.println(" Message from non static nested or
Inner class : " + message);
        }
    }


    // How to create instances of static and non-static nested
class
    public static void main(String... args){

        // creating instance of nested Static class
        Outer.MessagePrinter printer = new Outer.MessagePrinter();

        //calling non static method of nested static class
        printer.printMessage();

        // creating instance of a non static nested class or Inner
class

        // In order to create an instance of Inner class, you need
an Outer class instance

        Outer outer = new Outer(); //outer class instance for
creating non static nested class

        Outer.Inner inner  = outer.new Inner();

        //calling non static method of Inner class
        inner.display();

        // we can also combine the above steps in one step to
create an instance of Inner class
        Outer.Inner nonStaticIner = new Outer().new Inner();

        // similarly you can now call Inner class method
        nonStaticIner.display();
    }

}

Output:
Message from nested static class: HelloWorld
Message from non-static nested or Inner class: HelloWorld
Message from non-static nested or Inner class: HelloWorld
```

That's all on the **difference between the Static and non Static nested class in Java**. So far, we have only touched members Inner class and not discussed the other two types of

Inner class, like Local and Anonymous Inner classes. In this Java tutorial, we have seen What is nested static class is in Java and How to create instances of both nested static and non-static classes in Java.

In summary, it's easy to create instances of the nested static class as it doesn't require instances of Outer class while non-static nested class, e.g., Inner class, will always need an Outer class instance and can not exist without Outer class. If you have to choose between static vs. non-static class, then prefer static nested class if you can use that.