Anonymous & static classes

Dr. Rajesh Kumar Ojha
Asst. Prof., CSE, Silicon University

Anonymous Class

1. Anonymous class is a class which has no name.
2. Eventually the compile gives a name very similar to the parent class/interface. i.e. enclosingclassname$1
3. Anonymous class can be created when we want to create only one object of the class, where the name of the class becomes insignificant.
4. Anonymous classes can be also nested as inner class.
5. As it has no name so to create object as well as define it inline we need the reference of some base class.
6. The code in the next slide will give you a basic idea of anonymous class.

```
class A                          //Demo of Anonymous class
{
void show(){System.out.println("I am base class A");}
}
class AnonymousClass
{
        public static void main(String s[])
        {
        A a1=new A();
        a1.show();
        A a2=new A(){void show(){System.out.println("I am
anonymous class");}};
        a2.show();
        }
}
```

O/P: I am base class A

I am anonymous class

```
class A                          //Demo of Anonymous class with interface
{
void show();
}
class AnonymousClass
{
        public static void main(String s[])
        {
        A a1=new A(){void show(){System.out.println("I am
anonymous class");}};
        a1.show();
        }
}
```

O/P: I am anonymous class

Static Classes

- **The inner class defined with static key word is known as static class.**
- **Only inner classes can be declared as static.**

Differences between Static and Non-static Nested Classes

1. A static nested class may be instantiated without instantiating its outer class.
2. Inner classes can access both static and non-static members of the outer class.
3. A static class can access only the static members of the outer class.

```
class OuterClass {
        private static String msg = “Hi I am a static member";
        public static class NestedStaticClass {

        // Only static members of Outer class  is directly accessible in nested  static class
                public void printMessage()
                {       // Try making 'message' a non-static
                        // variable, there will be compiler error
                        System.out.println(
                                "Message from nested static class: " + msg);
                }
        }

        public class InnerClass {       // Non-static nested class  also called Inner class
                // Both static and non-static members  of Outer class are accessible in  this Inner
class
                public void display()
                {       // Print statement whenever this method is  called
                        System.out.println(“Message from non-static nested class: “ + msg);
                }
        }
}
```

```
class GFG {             // Main class
        public static void main(String args[])
        {               // Creating instance of nested Static class  inside main() method
                        OuterClass.NestedStaticClass printer = new OuterClass.NestedStaticClass();

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

                        // Creating Outer class instance for creating  non-static nested class
                        OuterClass outer = new OuterClass();
                        OuterClass.InnerClass inner = outer.new InnerClass();

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

                        // We can also combine above steps in one  step to create instance of Inner class
                        OuterClass.InnerClass innerObject = new OuterClass().new InnerClass();

                        innerObject.display(); // Similarly calling inner class defined method
        }
}
```

Thank you