

## Introducing Nested and Inner Classes

It is possible to define a class within another class; such classes are known as *nested classes*. The scope of a nested class is bounded by the scope of its enclosing class. Thus, if class B is defined within class A, then B is known to A, but not outside of A. A nested class has access to the members, including private members, of the class in which it is nested.

The most important type of nested class is the *inner* class. An inner class is a non-static nested class. It has access to all of the variables and methods of its outer class and may refer to them directly in the same way that other non-static members of the outer class do. Thus, an inner class is fully within the scope of its enclosing class.

The most important type of nested class is the *inner* class. An inner class is a non-static nested class. It has access to all of the variables and methods of its outer class and may refer to them directly in the same way that other non-static members of the outer class do. Thus, an inner class is fully within the scope of its enclosing class.

**// Demonstrate an inner class.**

```
class Outer
{
    int outer_x = 100;
    void test()
    {
        Inner inner = new Inner();
        inner.display();
    }
}

// this is an inner class
class Inner
{
    void display()
    {
        System.out.println("display: outer_x = " + outer_x);
    }
}
}
```

```
class InnerClassDemo
{
    public static void main(String args[])
    {
        Outer outer = new Outer();
        outer.test();
    }
}
```

**Output from this application is shown here:**

display: outer\_x = 100