

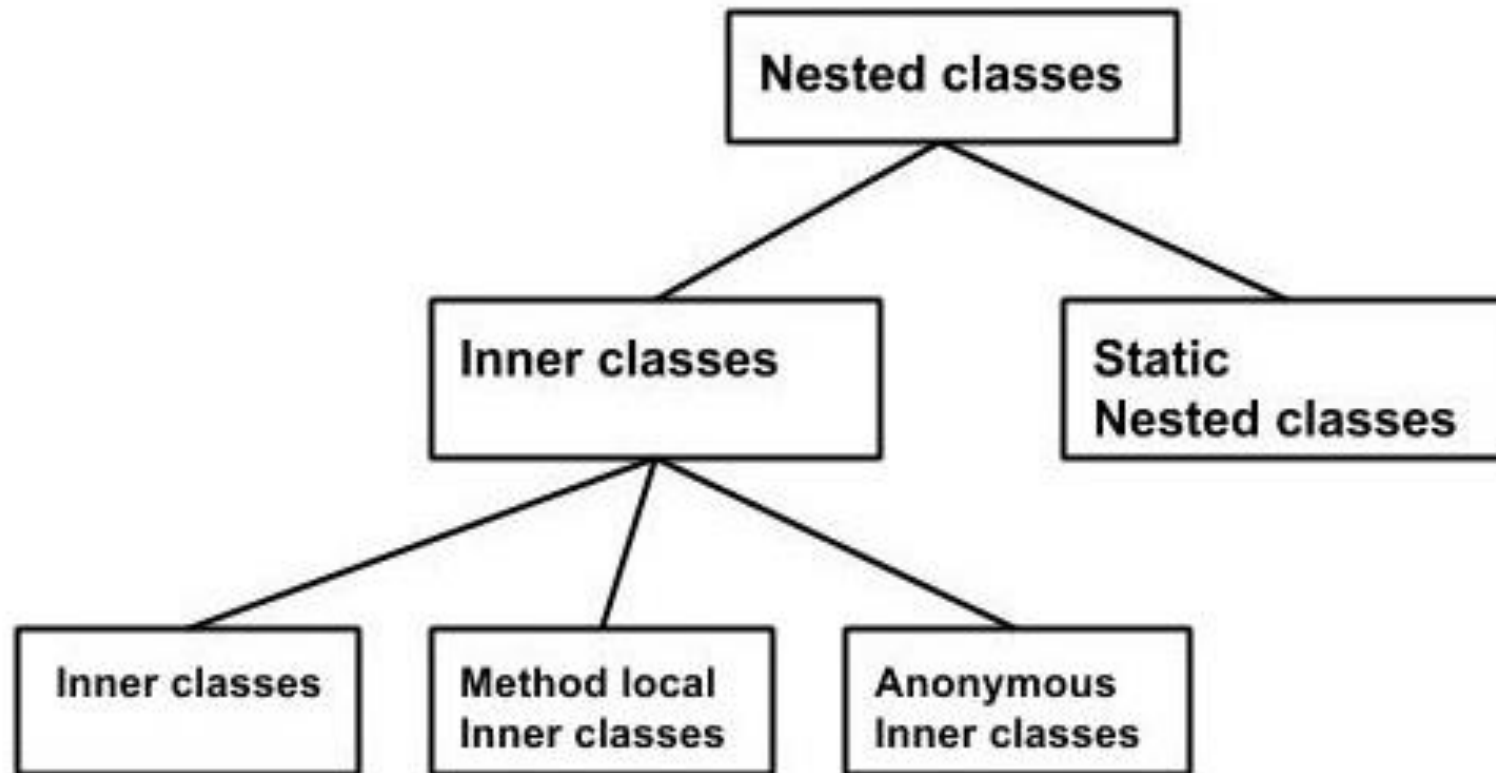
Java Programming

Sridhar A

Nested Classes



Nested Class Types



Nested Classes

```
class OuterClass {  
    ...  
    class NestedClass {  
        ...  
    }  
}
```

```
class OuterClass {  
    ...  
    static class StaticNestedClass {  
        ...  
    }  
    class InnerClass {  
        ...  
    }  
}
```

Why use Nested Classes?

- **It is a way of logically grouping classes that are only used in one place**
- **It increases encapsulation**
- **It can lead to more readable and maintainable code**

Shadowing

- Code Demo

Local Inner Class

- Local classes are classes that are defined in a *block*, which is a group of zero or more statements between balanced braces. You typically find local classes defined in the body of a method.

Anonymous Classes

- Anonymous classes enable you to make your code more concise.
- They enable you to declare and instantiate a class at the same time.
- They are like local classes except that they do not have a name.
- Use them if you need to use a local class only once.

Lambda Expressions



Lambda expressions let you express instances of single-method classes more compactly

Why Lambda?

- Code Example

Inline Values

```
String name = "foo";
```

```
double pi = 3.14;
```

```
aBlockOfCode = {  
    . . .  
    . . .  
}
```

Function as Value

```
aBlockOfCode = public void perform() {  
                System.out.print("Hello World!");  
            }
```

Function as Value

```
aBlockOfCode =      void perform() {  
                      System.out.print("Hello World!");  
                      }
```

Function as Value

```
aBlockOfCode = void () {  
    System.out.print("Hello World!");  
}
```

Function as Value

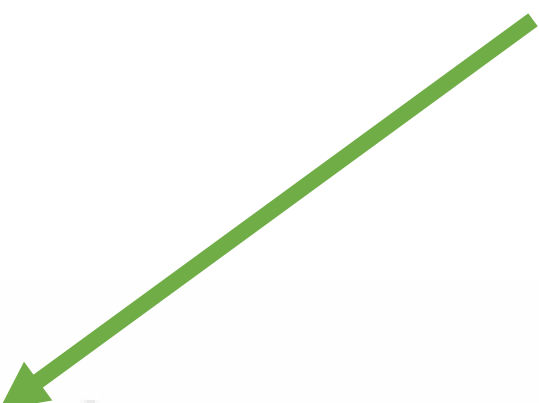
```
aBlockOfCode = void () {  
    System.out.print("Hello World!");  
}
```

Function as Value

```
aBlockOfCode = () {  
    System.out.print("Hello World!");  
}
```


Function as Value

```
aBlockOfCode = () -> {  
    System.out.print("Hello World!");  
}
```



Lambda Expressions

```
greetingFunction = () -> System.out.print("Hello world");

doubleNumberFunction = (int a) -> a * 2;

addFunction = (int a, int b) -> a + b;

safeDivideFunction = (int a, int b) -> {
    if (b == 0) return 0;
    return a / b;
};

stringLengthCountFunction = (String s) -> s.length();
```