CMSC 132: OBJECT-ORIENTED PROGRAMMING II

Nested Classes

Department of Computer Science

University of Maryland, College Park

Java Classes

- **Top level classes**
  - Declared inside package
  - Visible throughout package, perhaps further
  - Normally declared in their own file
    - Public classes must be defined in their own file
    - Not required for other classes (e.g., package)
    - **Example:** other package→Car.java
- **Nested Types**
  - Declared **inside** class or method
  - Normally used only in outer (enclosing) class
    - Can have wider visibility

4 Nested Classes

1. **Inner Class** - Only applies to classes. Will not have the keyword **static**
2. **Local Classes** - Only applies to classes. A class defined in a block of Java code (e.g., body of a function)
3. **Anonymous Class** - Only applies to classes. Local class without a name
4. **Static Class** - Can exist without outer class

Inner Classes

- **Description**
  - Class defined in scope of another class
  - Should not have any static members
- **Useful property**
  - Outer & inner class can directly access each other’s fields & methods (even if private)
  - Inside methods of outer class, use inner class as any other class
    - ic = new MyInnerClass()

Inner Class Example

- **Example (MyOuterClass and MyInnerClass are not Java reserved words):**

```
public class MyOuterClass {
    private int x;
    private class MyInnerClass {
        private int y;
        void foo( ) { x = 1; }   // accessing private field
    }
    void bar( ) {
        MyInnerClass ic = new MyInnerClass( );
        ic.y = 2;                // accessing private field
    }
}
```

Method Invocations

- Method invocations on inner class
  - Can be transparently redirected to outer instance
- Resolving method call on unspecified object
  - See if method can be resolved on inner object
  - If not, see if method can be resolved on corresponding instance of outer object
  - If nested multiple levels, keep on looking

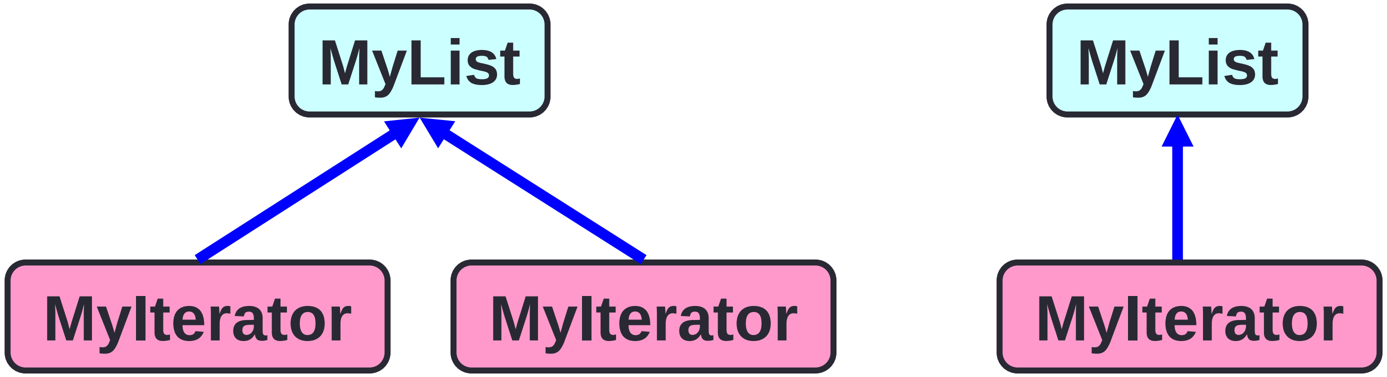
Accessing Outer Scope

- **Example**

```
public class MyOuter {                              // outer class
    int x = 2;
    private class MyInner {                         // inner class
        int x = 6;
        private void getX() {   // inner class method
            int x = 8;
            System.out.println( x );                // prints 8
            System.out.println( this.x );           // prints 6
            System.out.println( MyOuter.this.x );   // prints 2
        }
    }
}
```

Inner Class Link To Outer Class

- Inner class instance
  - Has association to an instance of outer class
  - Must be instantiated with an enclosing instance
    - **Inner class instance cannot exist without outer class instance**
  - Is tied to outer class object at moment of creation
    - Can not be changed



Inner Classes

- Useful for
  - Private helper classes
    - Logical grouping of functionality
    - Data hiding
  - Linkage to outer class
    - Inner class object tied to outer class object
      - E.g., wings of a plane
- Java Examples
  - Iterator for Java Collections
  - ActionListener for Java GUI widgets

Iterator Example

- Team class example

```
public class Team {
    private Player[] list;
    private int size;
    …
}
```

- Goal: Implement iterator for the class that allow us to access the players
- We will see different versions that implement the iterator. Using inner classes will simplify the iterator implementation

Team Class Example

- **Version 1**
  - No iterator
- **Version 2**
  - Iterator implemented without inner class
    - Illustrates problems of accessing private data of **Team** class
- **Version 3**
  - Iterator implemented using inner class

Team Class Example

- **Version 4**
  - Iterator implemented using inner class with class implementing Iterable<Player>
  - **Iterable** interface defines the method **Iterator<T> iterator()**
    - https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/Iterable.html
  - Returns an iterator over a set of elements of type T
  - Implementing this interface allows an object to be the target of the enhanced for loop "foreach" statement
- **Version 5**
  - Using a local class
- **Version 6**
  - First let’s go over anonymous inner classes
  - **Example:** anonymousClasses package
    - Using an anonymous class

Static Nested Class

- Similar to inner class, but declared as a static class
- No link to an instance of the outer class
- Can only access static fields & methods of the outer class
- **Can have an instance of the static nested class in your code even without having an instance of the outer class**
- Similar to a top-level class, but nested for packaging convenience
- **Example:** nestedStatic package

Additional Examples

- Inner class :
  - https://docs.oracle.com/javase/tutorial/java/javaOO/innerclasses.html
- Local class:
  - https://docs.oracle.com/javase/tutorial/java/javaOO/localclasses.html
- Anonymous class:
  - https://docs.oracle.com/javase/tutorial/java/javaOO/anonymousclasses.html