Some Philosophy

Nested and Inner Classes

The Singleton Pattern

Overview
called the Singleton Design Pattern.

There is a way to design a class to ensure such uniqueness property. It is

created throughout the life of a program.

How can we ensure that only one instance of EmptyListNode can be

set.

— The concept is akin to that of the empty set: there is only one empty

Conceivably, there is only one empty list in the "world".

Recall the EmptyListNode class that represents the "empty list".

The Singleton Pattern
class scope (i.e. static).

Note: The field instance and the method getInstance() are of

The following UML diagram describes the pattern:

The Singleton Pattern (cont.)
(Recall that we saw the Factory Pattern in Lab 2.)

In the special case, SingletonClass manufactures its own (unique) instance. The class SingletonClass is appropriately called a "factory." In this very

used to manufacture an instance, though unique, of the SingletonClass.

The method UniqueInstance() is called a "factory" method as it is

The Singleton Pattern (cont.)
Nested and Inner Classes

- The members of an inner class can access **ALL** members of the outer class.
- The members of an inner class can access only static members of the outer class.
- The members (i.e., fields, methods, classes) of a static (nested) class can access any other class just like any other class, a class defined inside of another class can
- The enclosing class is called the `outer class`.
- When it is non-static, it is called an `inner class`.
- When it is defined as static, it is called a `nested class`.

The rules for using such classes are similar to fields and methods.

* **Scope specifier:**
- be `public`, `protected`, `package private`, or `private`.

* **Access specifier:**
- be `public`, `protected`, `package private`, or `private`.

Besides fields and methods, a Java class can also contain other classes.
getter methods for the state.

its outer object (the context), there is no need to have setter and
impliment as inner objects. Since an Inner object has access to
* "In the state design pattern, the states of an object are often
  classes.
  Event listeners for Java GUI components are implement as Inner
  An inner object can be thought as an extension of the outer object.
* methods cannot do.

inner object of the outer object and perform complex tasks that simple
− Implementing hiding.
− Nested classes are used to create objects that have direct access to the
− Nested classes are used mostly to avoid name clash and to promote
  Usage:

**Nested and Inner Classes (cont.)**
its actual character value.

- When it is visible, it converts to a string as the string consisting of

- When it is hidden it converts to a string as "".

• In the Hangman game, a character in the target word can be either in

Example

February 11, 2000

Comp 212
All method calls in WordChar are delegated to its state.

1. Define a field in WordChar to reference an AState, its current state.
2. Define an abstract AState class as a private static nested class of WordChar.
3. Define inner classes HiddenState and VisibleState of WordChar as concrete subclasses of AState. AState and its concrete variants represent the states of a WordChar.
4. For a field in WordChar to reference an AState, its current state.

Example (cont.)
object-oriented programming design and concepts.

By implementing the hangman word as a system of cooperating

the singleton pattern.

This design makes use of the composite pattern, the state pattern, and

The UML diagram on the handout illustrates the above design.

Example (cont.)