# CIS 3145 Class Notes: Text Chapter 12

## Interface Concepts

**Objectives**

* Describe an Interface in the Java programming language
* Describe how the Interface compares to an abstract class
* Know how to create, implement and use an Interface
* Know how to implement and use a Cloneable interface

**Interfaces**

A class can ONLY inherit (“**extend**”) a **single** parent class in Java but it can “**implement**” many interfaces. {C++ does support *multiple inheritance* which Java does not.}

Also, an interface can inherit **many** otherinterfaces, and it uses the “**extends**” keyword to do so!

**What does an interface do?**

The interface defines a **standard** for what a class must be able to do if it has that interface, and it provides the standard **method** **signatures**. It also provides **public constants**.

The interface is a “**Promise**” (enforced by the compiler) to implement the defined methods in an object with that interface.

An interface can only have **static constants, methods** AND **Abstract Methods**, it cannot contain *variables* (instance or static), *instance constants* or *instance Methods*.

One exception is that *interfaces* can have a **default** instance method. All classes that *implements* this interface will have a print method.

Public interface Printable {

**Default** void print(){

System.out.println(toString());

}

}

**How is an interface used?**

The interface can be used to define a variable **data type**! Thus, we know that an object of a given interface type will have all the methods defined by that interface.

**Examples of interfaces**

Cloneable No Methods implemented -> Tagging interface

Comparable

EventListener

WindowListener

ActionListener

**Interface Implementation**

ALL methods in an interface are **public** and **abstract (**unless it is **default)**!

All fields in an interface are public, static, and final. But these three modifiers are implied so you don’t need to type them.

There is no code in the **abstract** interface method definitions, only the method **signature (**name, and parameter list)!

To use one or more interfaces a **class** will use the **implements** keyword.

However, one interface can **inherit** one or more other interfaces using the “**extends”** keyword. In contrast remember that a **class** can inherit **only one** other **class**.

**Product Viewer App**

The Product Viewer App can be broken down into the three logical layers of an information system: *Interface* layer, *Business Logic* layer, and *Data* layer.

Business Logic Layer

Interface Layer

ProductApp

Product Class

Data

Layer

Product Reader

ProductDB

The ProductApp creates an object for the ProductDB class but it does so by using the “ProductReader” interface data type. The advantage of the interface is that when the data layer is changed to a text file or database the code in the ProductApp does not have to change.