10

Abstract Classes and Interfaces

Last time

- inheritance
- polymorphism
- abstract classes (`abstract` keyword)

Objectives

- review abstract classes
- interfaces (`interface` and `implements` keywords)
- multiple interfaces
- event-driven programming
- GUI program: mouse and keyboard event handling

OOP Concepts

- Inheritance (`extends`)
- Polymorphism (upcasting to supertype)
- Abstract classes and methods (`abstract`)
- Interfaces (`interface, implements`)

OOP Syntax example

```
public class ClassName
     extends SuperclassName
     implements FirstInterface,
                SecondInterface, ...
{
    ...
}
```

Abstract classes and methods

- Keyword `abstract`
  - Use to declare a class `abstract`
  - Also use to declare a method `abstract`
  - Abstract classes contain one or more abstract methods

Interfaces

- **Interfaces**
  - **Declared using keyword `interface`**
  - **Contains only constants and `abstract` methods**
  - **Classes use `implements` to implement interfaces**

Interface objects and hierarchy

- **Objects of a class that implements the interface can also be thought of as objects of that interface**

- **Objects of any subclasses of the class that implements the interface can also be thought of as objects of the interface**

UML notation

- **UML representation of interfaces**
  - **Interfaces are distinguished from classes by placing the word "interface" in guillemets (« and ») above its name**
  - **Dashed pointer between a class and an interface is known as realization (class "realizes" the methods of an interface)**

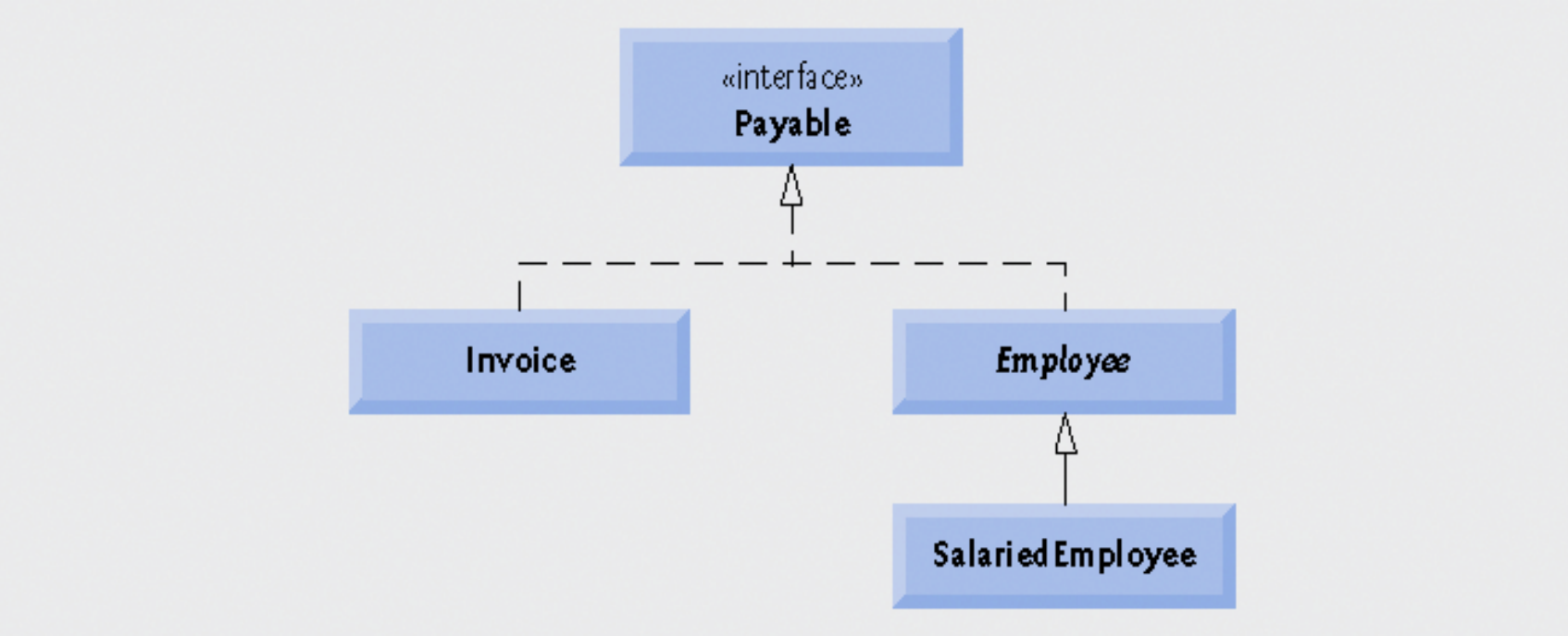

Fig. 10.10 | `Payable` interface hierarchy UML class diagram.

Interface	Description
`Runnable`	Implemented by any class for which objects of that class should be able to execute in parallel using a technique called multithreading (discussed in Chapter 23, Multithreading). The interface contains one method, `run`, which describes the behavior of an object when executed.
GUI event-listener interfaces	You work with Graphical User Interfaces (GUIs) every day. For example, in your Web browser, you might type in a text field the address of a Web site to visit, or you might click a button to return to the previous site you visited. When you type a Web site address or click a button in the Web browser, the browser must respond to your interaction and perform the desired task for you. Your interaction is known as an event, and the code that the browser uses to respond to an event is known as an event handler. In Chapter 11, GUI Components: Part 1, and Chapter 22, GUI Components: Part 2, you will learn how to build Java GUIs and how to build event handlers to respond to user interactions. The event handlers are declared in classes that implement an appropriate event-listener interface. Each event listener interface specifies one or more methods that must be implemented to respond to user interactions.
`SwingConstants`	Contains a set of constants used in GUI programming to position GUI elements on the screen. We explore GUI programming in Chapters 11 and 22.

Common interfaces of the Java API.

Multiple interface implementations

- **Class can implement as many interfaces as needs**
  - **Use a comma-separated list of interface names after keyword implements**
  - **Example:**

```
public class ClassName extends SuperclassName
implements FirstInterface, SecondInterface, ...
```

Mouse Event Handling

- **Mouse events**
  - `MouseEvent` **object (mouse state, coordinates, buttons, …)**
  - **Handled by** `MouseListener` **and** `MouseMotionListener`
  - **Interface** `MouseWheelListener` **declares method** `mouseWheelMoved` **to handle** `MouseWheelEvent`

MouseMotionListener interface methods

Methods of interface `MouseMotionListener`

`public void mouseDragged( MouseEvent event )`

Called when the mouse button is pressed while the mouse cursor is on a component and the mouse is moved while the mouse button remains pressed. This event is always preceded by a call to `mousePressed`. All drag events are sent to the component on which the user began to drag the mouse.

`public void mouseMoved( MouseEvent event )`

Called when the mouse is moved when the mouse cursor is on a component. All move events are sent to the component over which the mouse is currently positioned.

`MouseMotionListener` interface methods.

Keyboard Event Handling

- `KeyListener` **interface**
  - **For handling** `KeyEvent` **(keyboard events)**
  - **Declares methods:**
    - `keyPressed`
    - `keyReleased`
    - `keyTyped`

GUI Program: Paint