**抽象类**

在 Java 语言中使用 abstract class 来定义抽象类。如下实例：

Employee.java 文件代码：

/\* 文件名 : Employee.java \*/

public abstract class Employee

{

private String name;

private String address;

private int number;

public Employee(String name, String address, int number)

{

System.out.println("Constructing an Employee");

this.name = name;

this.address = address;

this.number = number;

}

public double computePay()

{

System.out.println("Inside Employee computePay");

return 0.0;

}

public void mailCheck()

{

System.out.println("Mailing a check to " + this.name

+ " " + this.address);

}

public String toString()

{

return name + " " + address + " " + number;

}

public String getName()

{

return name;

}

public String getAddress()

{

return address;

}

public void setAddress(String newAddress)

{

address = newAddress;

}

public int getNumber()

{

return number;

}

}

注意到该 Employee 类没有什么不同，尽管该类是抽象类，但是它仍然有 3 个成员变量，7 个成员方法和 1 个构造方法。 现在如果你尝试如下的例子：

AbstractDemo.java 文件代码：

/\* 文件名 : AbstractDemo.java \*/

public class AbstractDemo

{

public static void main(String [] args)

{

/\* 以下是不允许的，会引发错误 \*/

Employee e = new Employee("George W.", "Houston, TX", 43);

System.out.println("\n Call mailCheck using Employee reference--");

e.mailCheck();

}

}

当你尝试编译 AbstractDemo 类时，会产生如下错误：

Employee.java:46: Employee is abstract; cannot be instantiated

Employee e = new Employee("George W.", "Houston, TX", 43);

^

1 error