第七章　類別與物件產生

7-1 物件導向設計理念

7-1-1 軟體 IC – 物件

利用物件變數來描述真實環境，可能會超出原來環境的事實而不知曉。在許多情況下，所描述的真實現象需要有相當的限制，才不會超出原來的意義。譬如，真實環境會有下列限制：

(1) 日期限制：所表示的日期應該是 1 ~ 12 月，每月至多 31 天；如果某一表示日期的變數超出這個範圍，應該被拒絕才對。

(2) 身材限制：一般人的身高大多不會超過 3 公尺，體重也大多不會超過 300 公斤，如果超過這個範圍也應該被拒絕才對。

(3) 薪資限制：依照勞委會規定，勞工薪資應該不可以低於最低薪資，如果過低的話，應該拒絕才對。

(4) 年齡限制：一般『人』的年齡應該不會有負數，或則低於 150 歲，如果超過這個範圍，也應該拒絕才對。

當我們利用某些變數去描述上述現象時，希望能具有上述限制的功能。也就是說，描述真實環境的變數，本身需要有判斷處理的功能，當所描述的環境因素超過了真實現象，必須主動拒絕。

在許多情況下，僅拒絕不合理的變數產生，並不一定能滿足現實環境的需求，可能需要變數能自我調整，並使其合理化才是正當的做法。由上述的需求可以看出，變數不再僅是某一記憶體空間的內容，也不僅被動的填入某一數值內容就能滿足，它必須具有適當的處理能力。

如此一來，描述環境的變數則成為處理單元（Process），它是一個堅固的個體，裡面包含著適當的記憶體（變數）與程式（方法），並且針對某一種特殊功能而製作，我們就將它稱

為『物件』(Object)。利用物件所描述真實環境，與編寫程式的技巧，則稱為『物件導向』(Object-oriented)。

當物件被產生之後，就擁有自己的獨立運作記憶體空間，與產生物件的主程式記憶體空間並不相衝突。由前文可知，物件是能提供某一種特殊功能的獨立個體，類似組合電子電路的『積體電路晶片』(IC)。我們也可以取得現有的物件個體，組成一套系統功能較強的軟體套件，因此，吾人將物件個體稱之為『軟體 IC』。硬體方面的 IC 晶片可被組合一個較大、功能較強的電路板，當需要更複雜設備時，又可以組合多個電路板，成為一套功能更強的資訊設備。同樣的，軟體 IC 的物件個體也同樣具有這些功能；我們組合多個物件個體，成為一套功能較強的軟體套件，但基本上他還是具有物件個體的特質。我們還是可以組合多個物件導向的軟體套件，成為另一套功能更強的軟體系統。更重要一點，這些軟體 IC 並不需要自行開發，可多人共同開發，或經由購買專屬套件取得。簡單的說，利用物件導向技術所開發出來的物件個體，它可以像 IC 晶片般的流通。當然啦！物件個體還有繼承性與多形性特質，我們於第九章再來討論。

7-1-2 物件的成員

簡單的說，物件即是活動式(Activity)描述真實環境的個體，為了使它具有主動式功能，物件個體內必須具有相當的處理能力。圖 7-1 為物件個體 (Object body) 內容的抽象圖，包含有：

(1) 變數成員：描述物件所欲表示的現象，可能是有形、無形、真實或虛擬事件。

(2) **方法成員：**基本上，變數成員並不能直接被存取，必須透過方法成員來存取，否則會破壞變數成員的完整性。

(3) **動態記憶體空間：**當物件被執行時，可能需要的某些動態記憶體，當作區域變數使用。為了讓它能獨立運作，不可以與其他執行中的程式相衝突，必須特別給予保護。

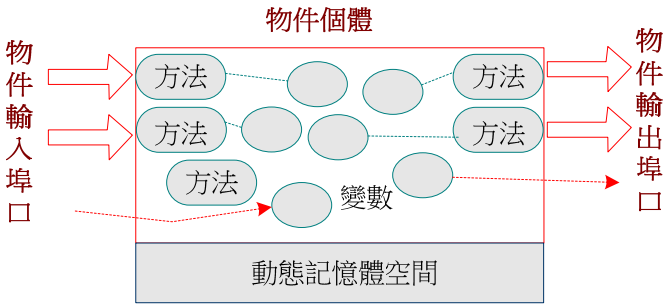

圖 7-1 物件個體的內容

7-1-3 變數成員的屬性

變數成員(Variable member)是提供物件所能描述真實環境的參數，可以由基本資料型態（如 int、float、char、array...等）、或其他物件所組成。依照變數可被存取的屬性區分為：

(1) 隱藏性變數（private 屬性）：此型變數僅允許物件內的方法成員存取，不允許外部程式擷取其內容。

(2) 公開性變數（public 屬性）：允許物件內或外部程式存取其內容。

(3) 共用性變數（static 屬性）：此類型變數不因物件產生而改變其內容。具有靜態變數的類別，產生多個物件後，還是使用同樣的靜態變數，又稱為『類別變數』。

圖 7-2 表示某一類別或物件執行當中可能出現的現象，假設某一類別導入（import）另一個類別，並由新類別產生另一個物件(object_1)。新的物件裡有 3 個變數成員，其中 v1 與 v3 成員為隱藏性變數（private）；外部程式無法直接存取，必須呼叫該物件的方法成員（method_1() 或 method()），透過它們存取 v1 或 v3 內容。另外，v2 是屬於公開性變數(非隱藏性)，物件外程式可以直存取其內容。

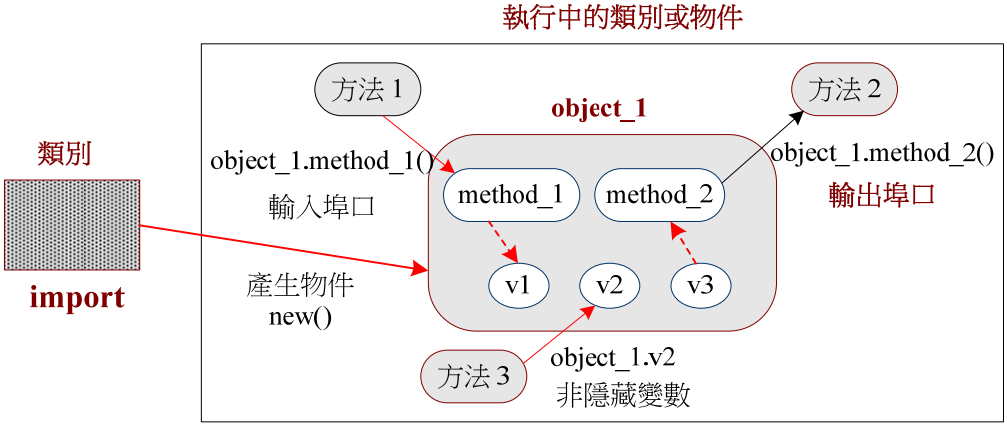

圖 7-2 變數成員的屬性

雖然物件所提供的特殊功能，大多是利用方法成員來實現。但隱藏性變數一定需透過某些方法成員，才可以外界溝通，因此稱提供此功能的方法為『物件輸出/輸入埠口』(如圖 7-1 所示)。物件方法提供填入物件變數內容功能者，稱之為『輸入埠口』；反之，提供攫取變數內容的方法，則稱之為『輸出埠口』。

7-1-4 方法成員的屬性

方法成員(Method member) 是物件個體內的函數，可執行某一特定功能的程式。最基本的功能是提供變數成員的輸入與輸出介面，亦可區分為：

(1) 公開性方法 (public 屬性) : 此類型的方法可任被所屬物件內或外部呼叫。

(2) 私有性方法 (private 屬性) : 僅允許所屬物件內其他方法呼叫。

(3) 保護性方法 (protected 屬性) : 允許所屬物件或原類別衍生出來物件的方法呼叫。

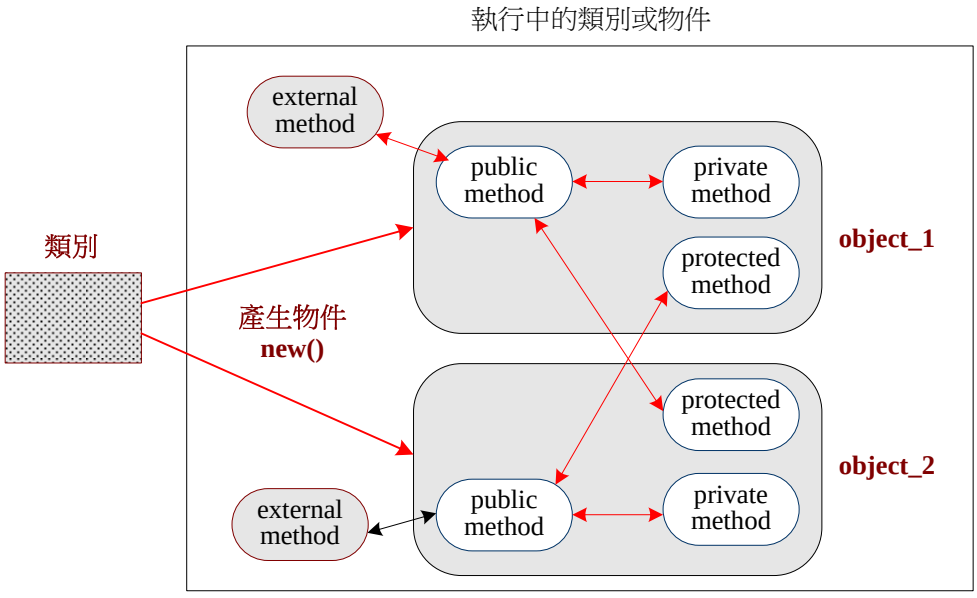

圖 7-3 方法成員的屬性

吾人以圖 7-3 來說明方法成員的屬性如何。假設某一執行中的類別經由某一類別產生了兩個物件 (object_1 與 object_2)，外部方法 (external method) 可以直接呼叫執行物件內的公開性方法(public method)，但無法呼叫私有性與保護性方法。然而私有性法(private method) 可以被相同物件內的其他方法呼叫；保護性方法 (protected method) 可以被由相同類別產生

物件內的方法呼叫。

7-1-5 類別與物件的關聯

簡單的說，類別（Class）就是物件的藍圖，或稱為描述檔。我們利用類別描述所期望建立物件的內容，需要時再利用類別產生物件（new 的功能）。就好像蓋房子一樣，類別即是設計房子的藍圖，利用此藍圖經由實地建造之後，便完成所設計的房子，該房子即是物件的意思。一份建築藍圖可以多次使用，蓋出許多相同樣子的房子；也就是說，一只類別可以產生多個名稱不同但功能相同的物件（如圖 7-4 所示）。

一份建築藍圖可經由增減功能成另一份建築藍圖，再利用新的建築藍圖去蓋另一種房屋；同樣的道理，一份類別可經由增加或修改功能成另一個新的類別，再利用新類別去產生另一種不同型態的物件，這就是類別的繼承性。

但建築圖與類別之間還是稍有不同，建築圖必須經過施工建設才能完成一間房屋才能居住；類別則不然，它不一定需要經過建設產生（new() 操作）才能使用，許多地方可以導入類別並直接執行，這方面本書預留在第九章介紹。本章僅介紹需經由類別產生物件後才可以執行。

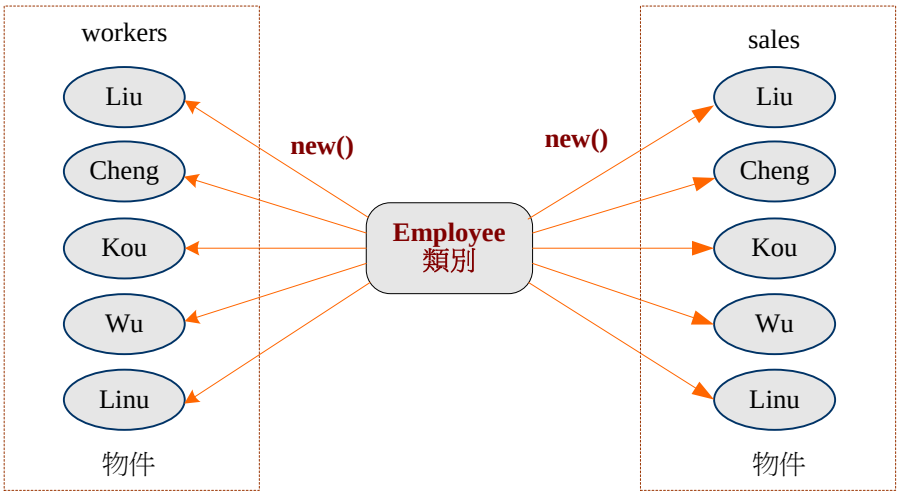

圖 7-4 類別與物件的關聯

7-2 類別的產生

7-2-1 類別宣告語法 - class

本書於第八章有稍稍介紹類別宣告的語法，但並不完整，這裡再重新詳細介紹，語法與範例如下：

類別及成員宣告語法：	範例：

[modifier] class class_name {
[modifier] declaration variable 1;
[modifier] declaration variable 2;
…..
[modifier] declaration method 1;
[modifier] declaration method 2;
….
}
	public class Employee {
	private int ID;
	String name;
	…….
	int getID() { …..}
	void setID(int i) { ….}
	……
	}

宣告類別包含類別本身（class body）、類別內的變數成員（class variable）、以及方法成員（class method）等三種元素，這三種元素都可分別宣告其屬性。

屬性有共通的表現方法，即是宣告語法前面有一個修飾字（modifier）選項（中括號 [ ] 表示選項的意思），表示該類別（或類別成員、類別方法）可以被存取的屬性宣告。類別與方法成員的修飾字有 public 與 private 等兩種，變數成員有 public、private 與 protected 三種屬性格式，說明如下：

(1) public： 公開性。如果類別或成員（方法或變數）被宣告成 public 型態，則表示它可以被任意呼叫使用。

(2) private：私有性。如果類別或成員（方法或變數）被宣告成 private 型態，則表示只允許類別內的方法才可以呼叫或使用。

(3) protected：保護性。如果變數成員（僅變數成員適用）被宣告成 protected 型態，則表示該成員可以被原類別所衍生出來的類別存取。

7-2-2 方法成員的宣告

如果類別內的方法是針對類別變數處理的話，則該類別所衍生的物件就成為一個很堅固的結構資料；如果是針對某一種特殊功能所設計方法的話，則該類別所衍生出來的物件，便成為專屬功能的函數了。由此可見，方法成員即是實現物件功能的主要藍圖。方法成員的宣告語法如下：

宣告方法成員語法：	範例：

[modifier] retrun_type method_name ([arg]{
….
method_body
…..
}
	int setPay(int pay) {
	if (pay <158000) {
	System.out.printf(“底薪太低\n”);
	retrun 0;
	}
	else
	payment = pay; return 1;
	}

重點說明如下：

(1) 修飾字（modifier）可能是 public 或 private，如沒有標明，則為 public。

(2) return_type（傳回值資料型態）可能是某一基本資料型態（如 int、char, ..)，或參考資料型態（如 String 或其他物件型態)。

(3) method_name 為該方法的名稱。

(4) 引數值（arguments）表示呼叫此方法時，傳遞給該方法數值的資料型態與變數。

(5) method_body 為該方法的主體實現，其中可能包含該主體內的變數宣告（區域變數）與其他程式控制（如 if/while, …)。

7-2-3 變數成員的宣告

在物件導向程式裡，變數成員代表著許多重大的意義，可能是描述某一真實物件的屬性，這些屬性都代表著某一特別的參數。簡單的說，類別變數的集合就好像 C 語言的結構變數一樣的功能。宣告語法與範例如下：

類別及成員宣告語法：	範例：
data_type variable_name; [modifier] data_type variable_name;	String name; private int ID;

重點說明如下：

(1) 修飾字（modifier）可能是 public、private 或 protected，如沒有標明，則為 public。

(2) data_type：變數成員的資料型態，可能是 int、float、double、char、String...等 或由另

一個類別產生的物件。

(3) variable_name：變數成員的名稱。

7-2-4 主方法的宣告 – main()

一個完整的程式，可能由多個類別所建構而成，然而每一個類別裡也許又包含了若干個方法，當這個程式被啟動時，到底是由哪一個類別中的哪一個方法開始執行？Java 程式與 C 程式語言都是由 main() 程序（Java 稱為方法），啟動開始執行。但 Java 程式有較特殊的地方，因為它是由多個類別所構成，必須指名是哪一個類別底下的方法，因此類別名稱必須與檔案名稱相同；範例如下：

```
//Hello.java
public class Hello {
        public static void main(String[] args) {
                System.out.println("Hello !! Your welcome");
                System.out.println("Java 程式設計 !! 歡迎您 ");
        }
}
```

重點說明如下：

(1) 檔案名稱（Hello.java）必須與類別名稱（Hello）相同。

(2) 類別的修飾字必須是 public，表示允許外部呼叫使用。

(3) main 方法的修飾字必須是 public static，其中 static 表示 main() 是類別方法（不必利用 new() 呼叫產生，容後介紹），並且可以任意被呼叫（public）。

(4) void 表示不會傳回任何值。

(5) String args[] 表示允許以 String 型態帶入引數。

(6) 也就是說 public static void main(String args[]) 標頭是缺一不可的。

7-3 物件的產生

7-3-1 物件產生敘述 – new()

設計好類別藍圖之後，接下來必須考慮如何將類別實現成為一個可執行運作的『物件』。像是蓋房子一樣，同一張建築藍圖可能因所需的人不同、所蓋的地理位不同、所建設的時間不同...等等不同因素，即使照圖蓋出外觀一樣的房子，內裝樣式卻可能出現很大的差別。由此可見，由同一份類別所產生的物件之間，也會因所面臨的狀況不同，產生外觀相同，內部大異其趣的物件。因此，任一時機內，由類別產生一個物件，只不過當時狀況的一個『驗例』(Instance)而已，由類別產生物件的過程稱為『驗例化』(Instantiate，某些地方稱為物件化)。利用 new () 函數呼叫，由類別產生一個物件，語法如下：

	物件產生語法	範　　例
物件宣告	class_name object_name;	Employee Liu;
物件產生	object_name = new class_name();	Liu = new Employee();
物件宣告並產生	class_name object1 = new class_name();	Employee Cheng = new Employee()
物件內變數	object_name.variable_name	Cheng.name
物件內方法	object_name.method_name	Cheng.setID()

重點說明如下：

(1) class_name object_name(如 Employee Liu): 宣告某一資料型態為 class_name 類別(如 Employee) 的物件，其名稱為 object_name (如 Liu)。其實宣告類別的語法，與宣告基本資料型態變數的方法很類似，如 int k 語句。

(2) object_name = new class_name() (如 Liu = new Employee()): 產生一個 class_name 型態 (如 Employee) 的物件，並填入 Liu 物件內。像是整數宣告 int k = 10 。

(3) 上述兩個敘述可整合一個敘述達成，則為 class_name object1 = new class_name()，譬如 Employee Cheng = new Employee ()。

上述範例中，Employee 為事先製作好的類別，並經過編譯成 Employee.class(Bytecode，中介程式)；而且須與原程式儲存放於同一目錄底下，否則必須利用 classpath 變數 (第一章有設定 classpath 說明)，標明 Employee.class 的所在位置。利用 new 產生的物件其特性如下：

(1) **變數成員表示**：經由類別產生的物件，該物件內的變數成員也繼承了原類別所規劃的屬性，可能是 public、private 與 protected 的存取限制。變數成員的表示方法為：

『object_name.variable_name』(如 Cheng.name)

(2) **方法成員表示**：物件內的方法成員承襲了原類別所規劃的屬性，可能是 public 或 private；方法成員表示為：

『object_name.method_name』(如 Cheng.setID())

7-3-2 範例研討：規劃通用型人事資料

（A）程式功能：Ex7_1.java、Employee.java

展鵬網路行銷公司需要一套較完整的人事資訊系統，該系統允許編輯員工資料，每一員工的描述屬性有：

(1) 員工代號 (ID)：整數。必須高於 1000，且低於 5000。

(2) 姓名 (name)：字串。

(3) 部門 (depart)：字串。

(4) 底薪 (payment)：整數。必須高於最低薪資 15800 元。

(5) 加班時數 (extra)：整數。最高 45 小時。

請先建立一只雛型程式，允許輸入員工資料，如輸入資料不符規定，則不予輸入並顯示錯誤原因。輸入完畢後印出該員工的薪資表，其中加班費計算方式為每小時 = (底薪 / (30 * 8)) * 1.5。期望操作介面如下：

(1) 編譯後，正常執行如下：

```
D:\Java2_book\chap7\Ex7_1>javac Employee.java      【編譯 Employee 類別】
D:\Java2_book\chap7\Ex7_1>javac Ex7_1.java         【編譯 Ex7_1 類別】

D:\Java2_book\chap7\Ex7_1>dir/b                    【Employee.class 須於同一目錄】
Employee.class
Employee.java
Ex7_1.class
```

```
Ex7_1.java

D:\Java2_book\chap7\Ex7_1>java Ex7_1
** 展鵬資訊 建立員工資料 **
請輸入 員工姓名 =>張大得
請輸入員工部門 = >資訊部
請輸入員工代號 =>1201
請輸入員工底薪 =>45000
請輸入員工加班時數 =>23
**** 列印員工薪資表 *****
代號    姓名      部門      本月薪資
1201     張大得   資訊部    49301
```

(2) 如果員工編號不在 1000~5000 範圍內，系統會拒絕操作並離開，如下：

```
D:\Java2_book\chap7\Ex7_1>java Ex7_1
** 展鵬資訊 建立員工資料 **
請輸入 員工姓名 =>張第一
請輸入員工部門 = >管理部
請輸入員工代號 =>134
需 1000 <員工代號 < 5000 範圍 !!
程式停止運作
```

(3) 如果員工薪資低於 15800，則系統會拒絕操作並離開，如下：

```
D:\Java2_book\chap7\Ex7_1>java Ex7_1
** 展鵬資訊 建立員工資料 **
請輸入 員工姓名 =>林添財
請輸入員工部門 = >管理部
請輸入員工代號 =>1203
請輸入員工底薪 =>12000
底薪不可低於 15800 元
```

```
程式停止運作
```

(4) 如果員工加班時數超過 45 小時，則系統會拒絕操作並離開，如下：

```
D:\Java2_book\chap7\Ex7_1>java Ex7_1
** 展鵬資訊 建立員工資料 **
請輸入 員工姓名 =>林森林
請輸入員工部門 = >製造部
請輸入員工代號 =>1208
請輸入員工底薪 =>24000
請輸入員工加班時數 =>56
勞基法規定加班時數不可超過 45 小時
程式停止運轉
```

（B）製作技巧研討：

物件的簡單應用，如同傳統語言的『結構變數』一樣，大多運用於描述環境的事實現象（或稱 Entity）。但物件不僅被動的描述 Entity 的屬性外，如果給予適當的處理，它可主動判斷『事項』是否已超過真實現象。以電子化人事系統為例，對於公司內員工薪資或工時計算大多不可以違反『勞工法』規定，譬如員工代號大多有一定格式、勞工最低薪資、每月最高超工時數、等等。

我們相信公司裡某些規定是適合所有員工，也就是員工資料制定完成之後，大多可以適用所有部門員工。因此，吾人依照展鵬公司的需求，製作了如圖 7-5 的 Employee 類別（Employee.java），其他應用系統也可直接引用，如此則可以整合公司內的員工資料。圖 7-5 是由 Employee 類別產生 worker 與 sales 員工物件的範例。設計完成員工資料類別之後，再編寫建立員工資料程式（Ex7_1.java）就不會困難了。

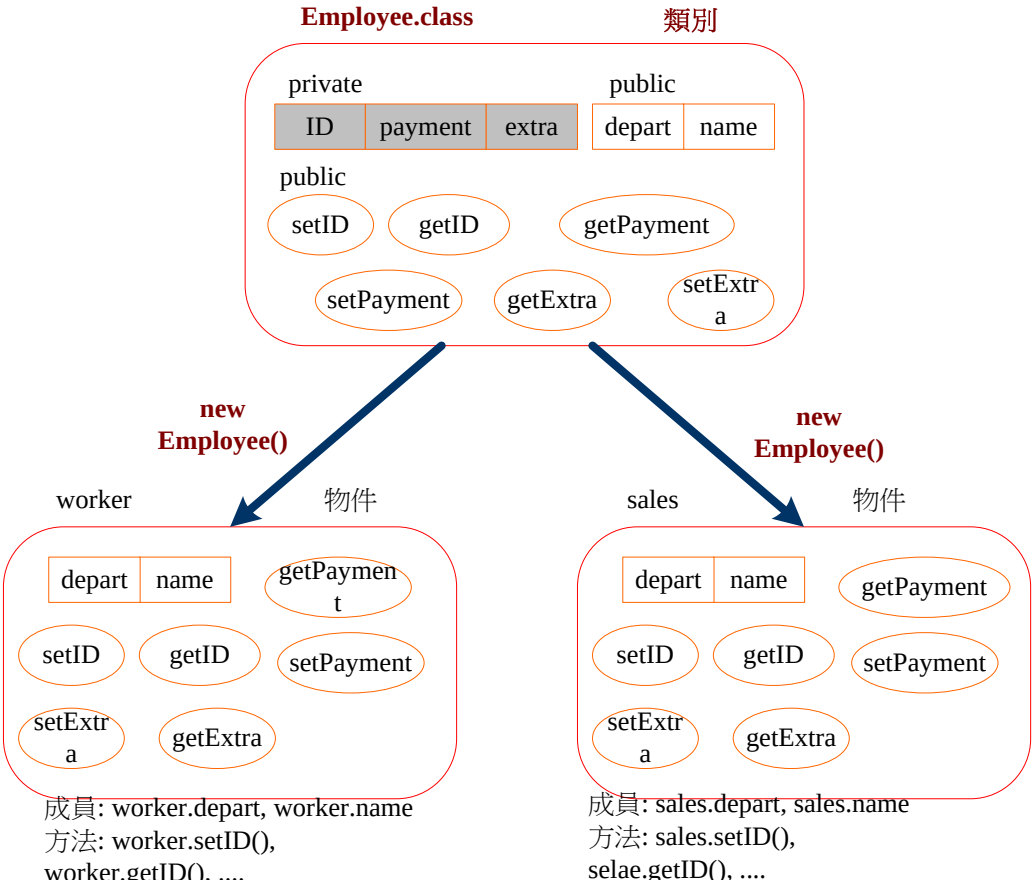

圖 7-5 員工資料的物件範例

（C） Employee.java 程式範例：

吾人利用 Employee.java 建構宣告員工物件的類別（Employee.class）；該類別內包含 5 個變數成員（其中 3 個是隱藏式變數），與 6 個存取隱藏變數的方法成員，然而它們即是該類別的輸入/輸出埠口。從外觀來看，Employee.class 抽象功能如圖 7-6 所示。各項功能如下說明：

(1) 屬於輸入埠口：

(a) int setID(int i)：設定員工代號，如 i 介於 1000~5000 之間則執行正常返回 1；否則返回 0。該方法是設定物件內變數成員，屬於輸入埠口之一。

(b) int setPayment(int pay)：設定員工底薪，如 pay 大於 15800，則執行正常並回傳 1；否則回傳 0，表示拒絕操作。屬於物件的輸入埠口之一。

(c) int setExtra(int ex)：設定員工加班時數，如 ex 小於 45 則執行正常並回傳 1；否則回傳 0。屬輸入埠口之一。

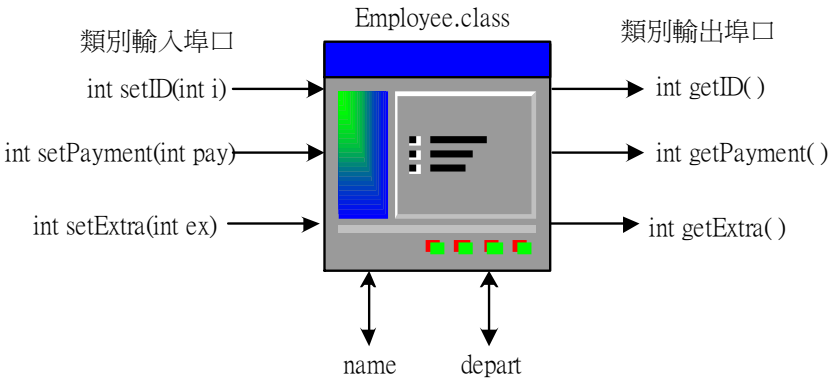

圖 7-6 Employee.class 類別架構

(2) 屬於輸出埠口：

(a) int getID()：取得員工代號。輸出埠口之一。

(b) int getPayment()：取得員工底薪。輸出埠口之一。

(c) int getExtra()：取得員工加班時數。輸出埠口之一。

(d) name：儲存員工姓名的變數，可任意存取。

(e) depart：儲存服務部門的變數，可任意存取。

(3) 屬於物件變數：

(a) 公開變數：有 name(員工姓名) 與 depart(服務部門)等 2 個變數，外部程式可以直接存取(讀或寫)。

(b) 私有變數：有 ID(員工編號)、payment(底薪) 與 extra(加班時數) 等 3 個變數，外部程式不可以直接存取，必須透過輸入/輸出埠口存取，如 setID()、getID()、、、等物件方法。

(4) Employee.java 程式範例，如下

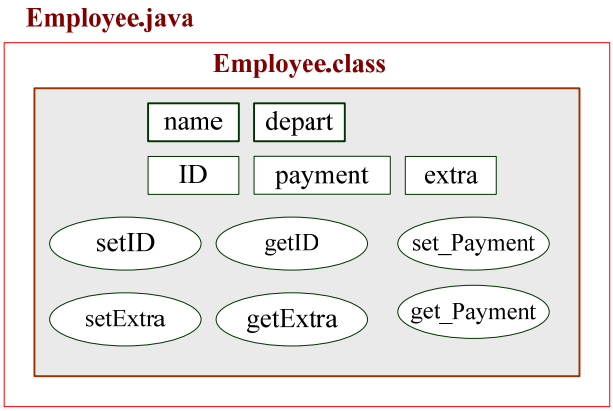

圖 7-7 Employee.java 程式架構

```
01  //Employee.java
02
03  class Employee {
04
05      /* 宣告物件成員 ***/
06
07      private int ID;                  // 員工代號, 介於 1000~5000
08      String name;                     // 員工姓名
09
10      String depart;                   // 所屬部門
11      private int payment;             // 底      薪 > 15800
12
13      private int extra;               // 加班時數 < 45
14
15      /*** 宣告物件方法 ***/
16
17      int setID(int i) {          // 設定員工代號方法, 1: 正常, 0: 錯誤
18              if((i>1000) && (i <5000)) {
19                      ID = i;
20                      return 1;
21              }
22              else {
23                      System.out.printf("需 1000 <員工代號 < 5000 範圍 !!\n");
24                      return 0;
25              }
26      }
27      int getID() {                    // 讀取員工代號方法
28              return ID;
29      }
30      int setPayment(int pay) {     //設定底薪方法, 1: 正常, 0: 錯誤
31              if (pay < 15800) {
32                      System.out.printf("底薪不可低於 15800 元\n");
33
34
```

```
35                  return 0;
36              }
37              else {
38                  payment = pay;
39                  return 1;
40              }
41          }
42          int getPayment() {          // 讀取底薪方法
43              return payment;
44          }
45          int setExtra(int ex) {      // 設定加班時數方法, 1: 正常, 0: 錯誤
46              if (ex > 45) {
47                  System.out.printf("加班時數不可超過 45 小時\n");
48                  return 0;
49              }
50              else {
51                  extra = ex;
52                  return 1;
53              }
54          }
            int getExtra() {            // 取得加班時數方法
                return extra;
            }
        }
```

程式重點說明：

(1) 第 6~10 行：宣告類別的變數成員，並指定其相關屬性（如 private）。

(2) 第 14~23 行：『int setID(int i) { ..}』。宣告 setID() 方法成員，該方法被呼叫時，會攜帶參數 i，如果該參數介於 1000 與 5000 之間，則這定員工代號，並回傳一個整數 1；否則拒絕設定，亦回傳 0。

(3) 第 24~26 行：『int getID() {…}』。宣告 getID() 方法成員，功能是回傳員工代號(ID)。

(4) 第 27~37 行：『int setPayment(int pay) { ..}』。功能是設定員工底薪的方法成員；當引數（pay）底薪低於 15800 元，該方法會拒絕輸入，並回傳一個整數 0；否則回傳整數 1。

(5) 第 38~40 行：『int getPayment() {…}』。 讀取員工底薪的方法成員。

(6) 第 41~50 行：『int setExtra(int ex) { ..}』。輸入員工加班時數的方法成員；當引數超過 45 時，會拒絕輸入並回傳 0；否則回傳 1。

(7) 第 51~53 行：『int getExtra() { … }』。讀取員工加班時數的方法成員。

（D）Ex7_1.java 程式範例

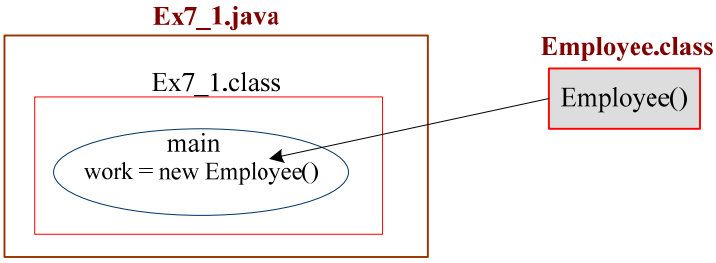

圖 7-8 Ex7_1.java 程式架構

製作完成 Employee 類別之後，吾人再編寫一主程式，引用該類別來產生工作員(worker)。針對 worker 物件輸入與輸出資料，證實 Employee 類別所提供的方法與變數成員是否能滿足所需。

```
01  //Ex7_1.java
02  /* 請建立一套人事資訊系統，該系統允許員工輸入資料，並印出該員工的薪資
03
04  表 */
05
06  import java.util.*;
07  // Employee.class 於同一目錄下
08
09  public class Ex7_1 {
10        public static void main(String args[]) {
11              Scanner keyin = new Scanner(System.in);
12              Employee worker = new Employee();
13              int id, pay, ex, pay1;
14
15              System.out.printf("** 展鵬資訊 建立員工資料 **\n");
16              System.out.printf("請輸入 員工姓名 =>");
17              worker.name = keyin.nextLine();
18              System.out.printf("請輸入員工部門 = >");
19              worker.depart = keyin.nextLine();
20
21              System.out.printf("請輸入員工代號 =>");
22
23
```

```
24              id = keyin.nextInt();
25              keyin.nextLine();
26              if (worker.setID(id)==0) {
27                      System.out.printf("程式停止運作 \n");
28                      return;
29              }
30
31              System.out.printf("請輸入員工底薪 =>");
32              pay = keyin.nextInt();
33              keyin.nextLine();
34              if (worker.setPayment(pay) == 0) {          // 判斷是否正常設定
35                      System.out.printf("程式停止運作 \n");
36                      return;
37              }
38
39              System.out.printf("請輸入員工加班時數 =>");
40              ex = keyin.nextInt();
41              keyin.nextLine();
42              if (worker.setExtra(ex) == 0) {                 // 判斷是否正常設定
43                      System.out.printf("程式停止運轉 \n");
44                      return;
45              }
46
47              System.out.printf("**** 列印員工薪資表 *****\n");
48              System.out.printf("代號    姓名    部門    本月薪資\n");
49              System.out.printf("%d\t", worker.getID());
50              System.out.printf("%s\t", worker.name);
51              System.out.printf("%s\t", worker.depart);
52              pay = worker.getPayment();
53              pay1 = (int)((double)pay/(30 * 8)) * worker.getExtra();
54              System.out.printf("%d\n", pay+pay1);
        }
}
```

程式重點說明：

(1) Employee.class 需存在於相同目錄下。

(2) 第 9 行：『Employee worker = new Employee();』。利用 Employee 類別產生 worker 物件，則該物件擁有原類別所宣告的方法與變數成員。

(3) 第 14 行：『worker.name = keyin.readLine();』。Worker.name 變數成員屬於公開性的（非隱藏性），可以直接存取。

(4) 第 21~24 行：『if (worker.setID(id)==0) { ..}』。呼叫 worker 物件的 setID() 方法成員，表示透過 setID() 將引數 id 值寫入 ID 變數內。執行後回傳是 0 的話，表示執行不正確。因 worker.ID 是隱藏性變數，無法直接存取，必須透過其他方法成員才行。

（E）直接存取隱藏性變數的結果

無人稍修改 Ex7_1.java 程式，測試直接存取 worker.ID、worker.extra 與 worker.payment 等隱藏性變數。修改部分程式內容：Ex7_1_1.java

```
……..
/* 驗證直接存取 private ID 的結果 */
            id = worker.ID;                  // 直接存取 private 屬性的變數成員
            System.out.printf("ID = %d\n", id);
/* 結束驗證 */
……..
```

其編譯結果如下：

```
D:\Java2_book\chap7\Ex7_1>javac Ex7_1_1.java
Ex7_1_1.java:27: error: ID has private access in Employee
            id = worker.ID;
                       ^
1 error
```

7-3-3 自我挑戰：驗證身分證字號

（A）程式功能：PM7_1.java、Customer.java

『真水水塑身美容中心』希望建立一套客戶資料管理系統，登錄客戶資料，針對每位客戶包含有下列資料：

(1) 身分字號（String SID）：須符合標準格式。

(2) 姓名（String name）：

(3) 性別（String sex）：僅提供女性服務。

(4) 年齡（int age）:

(5) 地址（String addr）:

請建立一個雛型客戶資料管理系統，測試系統規劃是否完備；該系統允許輸入客戶資料，再顯示是否正確。期望操作介面如下。

(1) 編譯後，正常執行如下：

```
D:\Java2_book\chap7\PM7_1>javac Customer.java      【編譯 Customer.java】
D:\Java2_book\chap7\PM7_1>javac PM7_1.java         【編譯 PM7_1.java】
D:\Java2_book\chap7\PM7_1>dir/b
Customer.class
Customer.java
PM7_1.class
PM7_1.java
D:\Java2_book\chap7\PM7_1>java PM7_1
** 真水水朔身中心 建立客戶資料 **
請輸入 客戶姓名 =>張美麗
請輸入身分字號 = >A277465487
請輸入性別 =>女
請輸入年齡 =>30
請輸入地址 =>高雄市鳥松區
=== 列印客戶資料 ===
身分證號          姓名   性別   年齡      地址
A277465487        張美麗   女        30        高雄市鳥松區
```

(2) 如果客戶身分證號錯誤，則系統會拒絕操作並離開，如下：

```
D:\Java2_book\chap7\PM7_1>java PM7_1
** 真水水朔身中心 建立客戶資料 **
請輸入 客戶姓名 =>劉美麗
請輸入身分字號 = >K123456723
這是偽造的 !!
程式停止運作
```

(3) 如果客戶不是女性，則系統會拒絕操作並離開，如下：

```
D:\Java2_book\chap7\PM7_1>java PM7_1
** 真水水朔身中心 建立客戶資料 **
請輸入 客戶姓名 =>劉大勇
請輸入身分字號 = >A277465487
請輸入性別 =>男
僅服務女性 !! 對不起
程式停止運作
```

（B）製作技巧提示

本系統最大的特點是，必須偵測所輸入的身分證字號是否正確，由這一點可觀察出來客戶所提供的資料是否偽造的。因此，需在客戶資料類別（Customer.class）必須提供此檢查功能。身份證號碼第一個字元 A～Z 表示地區，第二個字元 1（男性）或 2（女性）表示性別，後面緊接著 7 個數字（0～9），最後是檢查碼（0~9），總共 10 個字。

檢查方法如下：首先將英文字母以(A=10 B=11 C=12 D=13 E=14 F=15 G=16 H=17 J=18 K=19 L=20 M=21 N=22 P=23 Q=24 R=25 S=26 T=27 U=28 V=29 W=30 X=31 Y=32 Z=33 I=34 O=35）轉換成數字，與原有數字組成 11 個數字；接著，第 1 數乘以 1、第 2 數乘以 9、第 3 個數乘以 8 ，依此類推第 9 個數乘以 2，第 10 個數乘以 1，將上述的結果相加，總和的個位數（num），再被 10 減的結果（10-mun），與第 11 個數字相同，則表示正確。

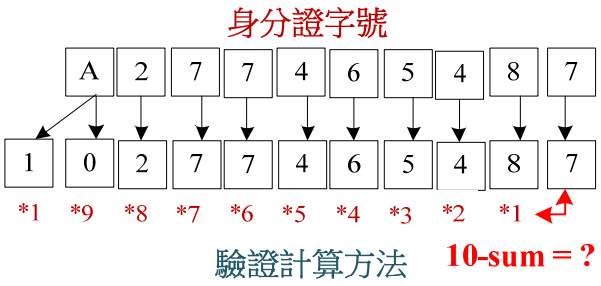

圖 7-8-1

本系統需要 Customer.java 與 PM7_1.java 兩只類別程式，前者規劃 Customer 類別使用；後者為主類別程式，本書僅提示 Customer.java 程式，另者請讀者自行編寫（請參考範例 Ex7_1.java）。

（C）Customer.java 程式範例

吾人針對系統要求登錄客戶哪些資料與相關限制，製作了 Customer 類別（Customer.java），其功能架構如圖 7-7 所示，各項功能如下：

(a) int setSID(String No)：設定客戶身份證字號方法。如果 No 字號符合標準格式，則執行正常，並回傳 1；否則拒絕操作並回傳 0。

(b) int setSex(String Sex)：設定客戶性別的方法。如果 Sex 為女性則執行正常，並回傳 1；否則拒絕操作回傳 0。

(c) String getSID()：取得客戶身份證字號的方法。

(d) String getSex()：取得客戶性別的方法。

(e) String name：儲存客戶姓名的變數，可任意存取。

(f) int age：儲存客戶年齡的變數，可任意存取。

(g) String addr：儲存客戶地址的變數，可任意存取。

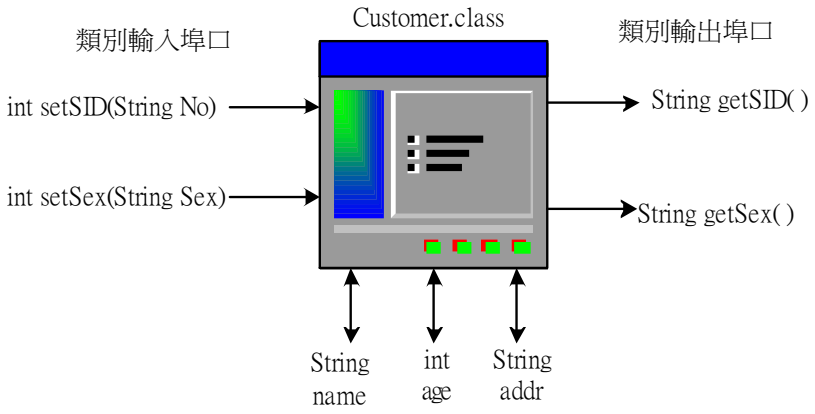

圖 7-9 Customer 類別功能架構

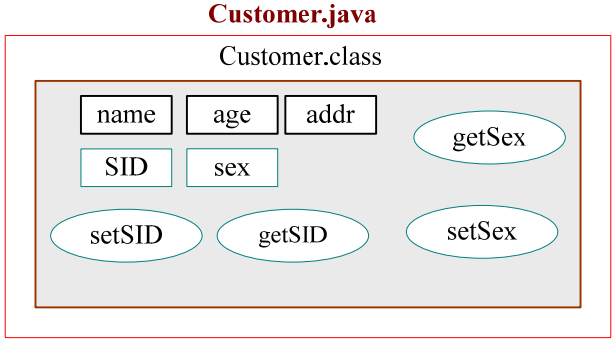

圖 7-10 Customer.java 程式架構

```
01 //Customer.java
02
03 import java.util.Scanner;
04 class Customer {
05       private String SID;            // 身分證字號, 須符合格式
06
07       String name;                   // 客戶姓名
08
09       private String sex;            // 性      別, 僅提供女性
10
11       int age;                       // 年      齡
12
13       String addr;                   // 地      址
14
15
16       int setSID(String No) {     // 設定身分字號方法, 1: 正常, 0: 錯誤
17             String[] dig1 = new String[10];     // 儲存身分證號字元
18
19             int[] dig2 = new int[11];           // 儲存轉換後數字
20             int temp;
21
22             if (No.length() != 10) {
23                   System.out.printf("需 10 個字元, 請重新輸入 !!");
24                   return 0;
25             }
26             Scanner s = new Scanner(No).useDelimiter("");
27             for(int I=0; I<10; I++)
28                   dig1[I] = s.next();
29
30             temp = Character.getNumericValue(dig1[0].charAt(0));
31             if (temp == 18)       // 字母 I, 設定為 34
32                   temp = 34;
33             else if (temp == 24)  // 字母 O, 設定為 35
34                   temp =35;
35             else if((temp>18) || (temp < 24))   // J ~ N 前進 1 個數
36                   temp = temp – 1;
37             else if (temp > 24)
38                   temp = temp – 2;              // P ~ Z 前進 2 個數
39
40             for(int I=1; I<10; I++)
41                   dig2[I+1] = Integer.parseInt(dig1[I]);
42             dig2[0] = temp / 10;
43             dig2[1] = temp %10;
44
45             int sum = dig2[0];          // 第 1 個數字乘以 1
46
```

```
47              int k = 9;
48              for(int I=1; I<10; I++) {            // 第 2 ~ 9 數字分別
49
50                      sum = sum + dig2[I] * k;     // 乘以 9 ~ 1 累加
51                      k = k-1;
52              }
53
54              int check = 10 –(sum % 10);       // 10 – 個位數
55
56              if (check == dig2[10]) {          // 是否與第 11 數字相同
57                       SID = No;
58                       return 1;
59              }
60              else {
61                      System.out.printf("這是偽造的 !! \n");
62
63                      return 0;
64              }
65       }
66
67      String getSID() {                    // 讀取員工代號方法
68              return SID;
69      }
70      int setSex(String Sex) {     //設定性別方法, 1: 正常, 0: 錯誤
71
72              if (Sex.equals("女")) {
73                      sex = Sex;
74                      return 1;
75              }
76              else {
                        System.out.printf("僅服務女性 !! 對不起 \n");
                        return 0;
                }
        }
        String getSex() {        // 取的客戶性別方法
                return sex;
        }
}
```

程式重點說明：

(1) 第 17~20 行：『if (No.length() != 10) { …}』。檢視輸入身證號(No)長度是否 10 個字元，如不是則拒絕操作，中斷並回傳 0。

(2) 第 21~23 行：『Scanner s = new … 』。將輸入身份證字號（No）各個字元取出，並分別存入 dig[] 陣列中。

(3) 第 25~34 行：『temp = Character.getNumericValue(..) …』。檢視身份正字號的第一個字母（A ~ Z），並轉換成相對應數字。

(4) 第 36~39 行：將其餘身份證號碼，與第一個字母相對應的數字，分別依序存入 dig2[] 陣列中。

(5) 第 41~46 行：累加計算 dig2[] 陣列內的數字。

(6) 第 48 行：『int check = 10 –(sum % 10);』。10 減掉累積後的個位數。

(7) 第 49~58 行：『if (check == dig2[10]) {…}』。與第 11 個數字比對是否相同，如是的話，格式相符；否則可能該身份證字號是偽造或輸入錯誤。

（C）PM7_1.java 程式提示

請讀者自行編寫，程式架構如圖 7-9 所示。

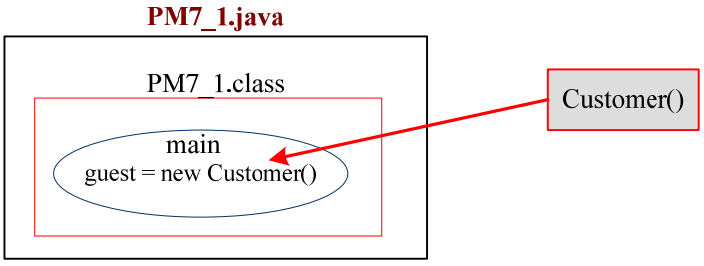

圖 7-11 PM7_1 程式架構

7-4 靜態變數的應用

7-4-1 類別變數 - static

基本上，經由類別產生若干個物件後，各個物件都屬獨立性個體，有自己的記憶體空間、變數成員與方法成員。但某些情況下，我們還是希望同一類別所產生的物件們，之間保留有一些連帶關係。讓物件之間保留關係，最簡單的方法的讓物件之間享有共同的變數成員，此型態變數稱為『靜態變數』（Static variable）或『類別變數』（Class variable）。類別實體內宣

告某一個靜態變數（如 static int a;）之後，無論該類別產生多個物件（如 x、y、z...等），靜態變數都指向同一變數（如 x.a、y.a、z.a）；也就是說，改變任一物件的靜態變數（如 x.a）的內容，則其他靜態變數的內容也隨之改變（如 y.a 或 z.a）；如圖 7-8 所示。

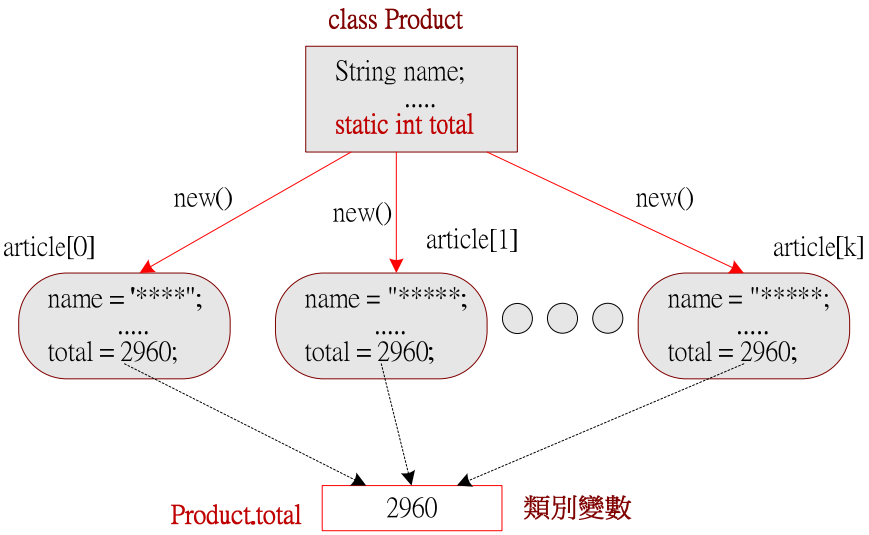

圖 7-12 靜態變數的特性

靜態變數的宣告語法如下：

```
class product {
    …..
    static int total;
}
```

重點說明如下：

(1) 宣告類別變數必須加入 static 關鍵字。

(2) 類別變數同樣具有 public、private 與 protected 屬性。

(3) 宣告範例如下：public static int a、private static String name...等。

7-4-2 範例研討：智慧型庫存管理系統

（A）程式功能：Ex7_2.java

許多經理人都想盡辦法降低庫存量與增加銷售量，來增加公司的獲利，但兩者其實互相衝突。當庫存量不足時，可能會嚴重影響到銷售的順暢性，庫存量太高則會積壓公司資金，

如何掌握最恰當的庫存量，實為考驗經理人能力的關鍵，最簡單的方法是隨時掌握目前庫存金額多寡。因此，『春嬌超商連鎖公司』期望建立一套智慧型的庫存管理系統。

該系統允許選擇商品及進貨或出貨數量，進貨用正號(如 +10); 出貨用減號(如 -10)。庫存檔案儲存於 product.data 內，每一商品包含四個欄位：{品名、單價、庫存量、金額、總庫存金額} (如 可口奶滋)，其中 "金額" 為該項產品庫存金額。"總庫存金額" 為所有庫存量的總金額；進出任何商品後，總庫存金額會登錄目前所有金額多寡。除了能滿足一般庫存系統的功能外，也能讓經理人隨時了解目前庫存金額多寡。期望操作介面如下：

```
D:\Java2_book\chap7\Ex7_2>javac Ex7_2.java

D:\Java2_book\chap7\Ex7_2>dir/b
Ex7_2.class                              【主程式類別】
Ex7_2.java                               【主程式檔案】
Product.class                            【產品類別】
Product.data                             【產品資料檔案】

D:\Java2_book\chap7\Ex7_2>java Ex7_2
****  未登錄前庫存量  ***
品名            單價     庫存量   金額     總庫存金額
可口奶滋        20       50       1000     2780
黑松汽水        12       40       480      2780
味全鮮乳        15       50       750      2780
頻果西打        8        80       640      2780

(1) 可口奶滋     (2) 黑松汽水     (3) 味全鮮乳     (4) 頻果西打
          請輸入貨品編號(或大於 5 離開) =>4
           進出貨數量 =>20
品名            單價     庫存量   金額     總庫存金額
可口奶滋        20       50       1000     2940
黑松汽水        12       40       480      2940
味全鮮乳        15       50       750      2940
頻果西打        8        100      800      2940
```

```
總庫存金額(product.total) = 2940
```

（B）製作技巧研討：

吾人將 Product 類別中庫存總金額宣告成類別變數 (static int total;)，當任何一樣產品變更此內容時，所有產品的 total 值隨之改變。因此，查閱任何一筆資料都可以知道目前總庫存金額多寡。

接著，吾人必須利用編輯工具(如 notepad ++) 建立一個 product.data 庫存資料檔，如下圖所示。進入系統後，它讀取庫存資料。

```
D:\Java2_book\chap7\Ex7_2>type product.data
可口奶滋        20      50      1000    2780
黑松汽水        12      40      480     2780
味全鮮乳        15      50      750     2780
頻果西打        8       80      640     2780
```

（C）程式範例：

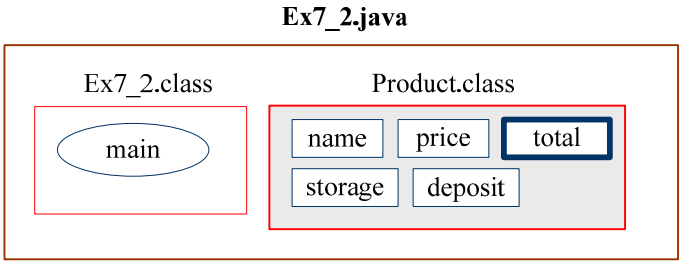

圖 7-13 Ex7_2 程式架構

```
01  //Ex7_2.java
02
03  import java.io.*;
04  import java.util.Scanner;
05  class Product {
06          String name;              //產品名稱
07
08          int price;               // 單價
09
10          int storage;             // 庫存量
11
          int deposit;               // 金額
```

```
12          static int total;         //  庫存總金額(類別變數)
13  }
14
15  public class Ex7_2 {
16      public static void main(String args[]) throws IOException {
17          Product[] article = new Product[20];
18          Scanner keyin=new Scanner(System.in);
19          String file = "Product.data";
20          String inData;
21          int k=0, select, count;
22          File fileID = new File(file);
23          if (fileID.exists()) {
24              BufferedReader data = new BufferedReader(new
25                                                  FileReader(fileID));
26              while ((inData=data.readLine()) != null) {
27                  Scanner s = new Scanner(inData).useDelimiter("\t");
28                  article[k] = new Product();
29                  article[k].name = s.next();
30                  article[k].price = s.nextInt();
31                  article[k].storage = s.nextInt();
32                  article[k].deposit = s.nextInt();
33                  article[k].total = s.nextInt();
34                  k = k +1;
35              }
36              data.close();
37          }
38          else{
39              System.out.printf("%s file not existed\n", file);
40              return;
41          }
42          System.out.printf("****  未登錄前庫存量  ***\n");
43          System.out.printf("品名\t\t 單價\t 庫存量\t 金額\t 總庫存金額\n");
44          for (int i=0; i<k; i++) {
45              System.out.printf("%s\t%d\t%d\t%d\t%d\n", article[i].name,
46                                  article[i].price, article[i].storage,
47                                  article[i].deposit, article[i].total);
48          }
49          System.out.printf("\n");
50          for (int i=0; i<k; i++)
51              System.out.printf("(%d) %s\t", i+1, article[i].name);
52          System.out.printf("\n");
53          System.out.printf("\t 請輸入貨品編號(或大於  5  離開) =>");
54          select = keyin.nextInt();
55          if (select < 5) {
56
57
```

```
58                System.out.printf("\t 進出貨數量 =>");
59                count = keyin.nextInt();
60                int s1 = select-1;
61                int m1 = article[s1].price * count;
62                article[s1].storage = article[s1].storage + count;
63                article[s1].deposit = article[s1].deposit + m1;
64                article[s1].total = article[s1].total + m1;
65          }
66          System.out.printf("品名\t\t 單價\t 庫存量\t 金額\t 總庫存金額\n");
67          for (int i=0; i<k; i++) {
68                System.out.printf("%s\t%d\t%d\t%d\t%d\n", article[i].name,
69                                          article[i].price, article[i].storage,
70                                          article[i].deposit, article[i].total);
71          }
72          System.out.printf("總庫存金額(product.total) = %d\n", Product.total);
73    }
}
```

（D）程式重點分析

(1) 第 21~40 行：系統啟動後，立即將 Product.data 檔案內所儲存的庫存資料讀入，並儲存於 article[] 物件陣列內，其中利用變數 k 紀錄有多少筆資料。

(2) 第 41~48 行：將 article[] 陣列內容顯示於螢幕上。

(3) 第 49~51 行：將陣列 article[] 內每一筆資料的產品名稱（article[i].name）印出，讓操作者選擇登錄哪一樣產品。

(4) 第 53~63 行：計算操作者所登錄產品的進出貨。

(5) 第 64~71 行：輸出登錄後的庫存資料。

7-4-3 範例研討：停車場管理系統

（A）程式功能：Ex7_3.java

高雄市政府公告『前金立體停車場管理系統』招標事項，該系統需具有下列功能：(假設範例)

(1) 顯示目前停車車輛：顯示目前每部車子的停車票號碼與進場時間。

(2) 車輛進入取票：車輛進入停車場時，會給予車子一個停車票，其中包含停車票號碼與進場時間。停車號碼由 1 開始累增。

(3) 車輛出場繳費：車輛離開時，需繳交停車票；系統搜尋該車輛紀錄時間，並計算停車費用 (每 30 分鐘 20 元)。

(4) 還未安裝計時時鐘，時鐘的時間由人工輸入設定。

參與投標者須建立一套模擬系統，除了可提供上述功能之外，為增加現場臨場感，需增加一項設定目前時間的功能，作為計算停車費的依據。期望雛型模擬系統具有下列操介面：

(1) 編譯後，正常執行如下：

```
D:\Java2_book\chap7\PM7_1>javac Customer.java      【編譯 Customer.java】
D:\Java2_book\chap7\PM7_1>javac PM7_1.java         【編譯 PM7_1.java】
D:\Java2_book\chap7\Ex7_3>javac carTicket.java     【編譯 carTicket.java】
D:\Java2_book\chap7\Ex7_3>javac Ex7_3.java         【編譯 Ex7_3.java】

D:\Java2_book\chap7\Ex7_3>dir/b                    【產生相關檔案 4 個類別】
Cars.class                                         【停車場車輛物件】
carTicket.class                                    【停車票物件】
carTicket.java
Clock.class                                        【公用計時物件】
Ex7_3.class                                        【主類別】
Ex7_3.java
```

(2) 主選單有 5 個選項，選項 1 為顯示目前停車場各輛車的進場時間、選項 2 為模擬計時器，利用人工輸入目前時間、選項 3 是車輛進入停車場，取票與開始計時、選項 4 是車輛離開時，繳費與刪除，如下：

```
D:\Java2_book\chap7\Ex7_3>java Ex7_3
** 高雄市前金立體停車場 管理系統 目前時間 0 時 0 分 **
(1) 顯示目前停車        (2) 設定目前時間
(3) 車輛進入取票        (4) 車輛出場繳費
(5) 離開系統
```

```
目前車輛 0 請輸入工作選項 =>
```

(3) 首先設定目前時間(選項 2)，如設定目前時間是 8 時 20 分，如下：

```
目前車輛 0 請輸入工作選項 =>2
請輸入目前時間(時/分) =>08/20

** 高雄市前金立體停車場 管理系統 目前時間 8 時 20 分 **
(1) 顯示目前停車          (2) 設定目前時間
(3) 車輛進入取票          (4) 車輛出場繳費
(5) 離開系統
```

(4) 當車輛進入時，會自動產生停車號碼，與登錄當時時間，如下：

```
目前車輛 0 請輸入工作選項 =>3
請按<Enter>鍵取票 (票號：1) =>
```

(5) 如進入 3 輛車後，顯示目前停車場車輛(選項 1)，如下：

```
目前車輛 3 請輸入工作選項 =>1
停車票號                    進入時間
1                   8 時 20 分
2                   9 時 10 分
3                   9 時 40 分

** 高雄市前金立體停車場 管理系統 目前時間 9 時 40 分 **
(1) 顯示目前停車          (2) 設定目前時間
(3) 車輛進入取票          (4) 車輛出場繳費
(5) 離開系統
目前車輛 3 請輸入工作選項 =>
```

(6) 當 2 號車欲離開時，其操作如下(選項 4)：

```
目前車輛 3 請輸入工作選項 =>4
```

```
請出示停車票 =>2
請繳交 20 停車費(按<Enter>鍵) =>

** 高雄市前金立體停車場 管理系統 目前時間 9 時 40 分 **
(1) 顯示目前停車          (2) 設定目前時間
(3) 車輛進入取票          (4) 車輛出場繳費
(5) 離開系統
目前車輛 2 請輸入工作選項 =>
```

（B）製作技巧提示：

製作此系統時需考慮車輛進出口並不一定會相同，即是系統可能由：入口（車輛進入取票功能）、出口（車輛出場繳費功能）、與管理處（顯示目前停車功能）等 3 個地方同時操作。因此需將儲存停車資料檔案設定成共享資料，為了方便系統設計，我們採用物件導向設計理念，圖 7-9 為資料流程概念圖。首先我們宣告停車票類別（carTicket），描述每張停車票的內容：ticket 與 enterTime[2]，前者為停車票號碼、後者為車輛進場時間（時與分）。另外，製作一個停車票紀錄類別 Cars，期望它是共用屬性，因此宣告成靜態類別，所有程式都可直接存取。Cars 類別包含：car[] 與 number，前者紀錄每張停車票的資料，後者紀錄目前停車場的車輛數目。本系統也須設定一個靜態時間類別 Clock，紀錄目前時間。

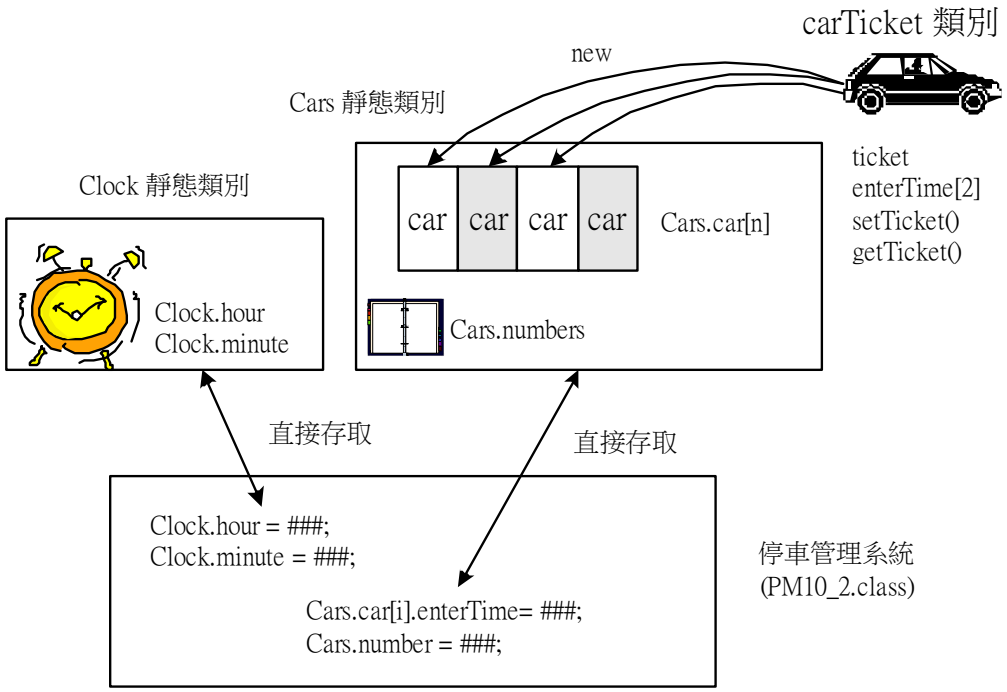

圖 7-14 範例 Ex7_3 資料流程

（C）相關類別設計（carTicket.java）

依照上述的規劃，我們需要 3 個類別，將其製作於 carTicket.java 檔案，編譯後會分別產生下列 3 個類別檔案（如圖 7-9 所示），功能如下：

(1) carTicket.class：停車票類別。功能是記錄停車票號碼，以及車輛進場時間；成員如下：

(a) int setTicket()：設定停車票號碼的方法。

(b) int getTicket()：取得停車票號碼的方法。

(c) enterTime[]：儲存車輛進入時間的變數，可任意存取。

(2) Cars.class：儲存停車票的類別變數，包含有：

(a) static carTicket[] car：進入車輛的停車票記錄（靜態變數）。由此記錄可知該車輛何時進入停車場，可做收費的依據。

(b) static int number：紀錄目前停車場的車輛數（靜態變數）。

(3) Clock.class：主時鐘的類別變數。記錄目前時間，包含有：

(a) Static int clock：目前幾點鐘（時數，靜態變數）。

(b) Static int minute：目前幾分鐘（分鐘數，靜態變數）。

(4) carTicket.java 原始檔案如下所示：

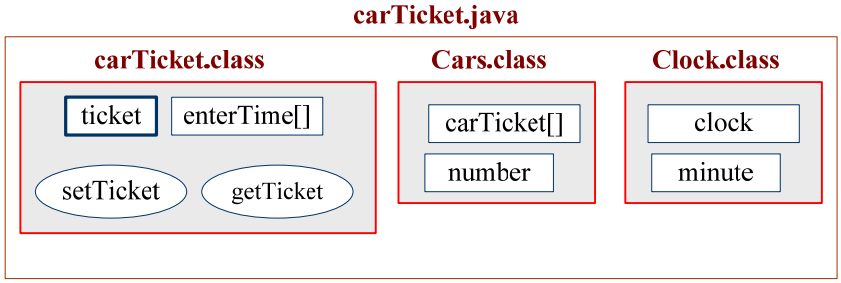

圖 7-15 carTicket.java 程式架構

```
01  //carTicket.java
02
```

```
03 /* 停車票  */
04
05 class carTicket {
06       private static int count=0;
07       private int ticket;            // 停車票號碼
08       int[] enterTime = new int[2];          // 進入停車場時間
09
10
11       int setTicket() {                        // 設定停車票號碼
12             count++;
13             ticket = count;
14             return ticket;
15       }
16       int getTicket() {                        // 取得停車票號碼
17             return ticket;
18       }
19 }
20 /* 目前停放車輛 */
21 class Cars {
22       static carTicket[] car = new carTicket[100];
23       static int number;
24 }
25 /*公眾計時器 (利用設定輸入) */
26 class Clock {
27        static int hour;
         static int minute;
       }
```

（D）主程式範例（Ex 7_3.java）

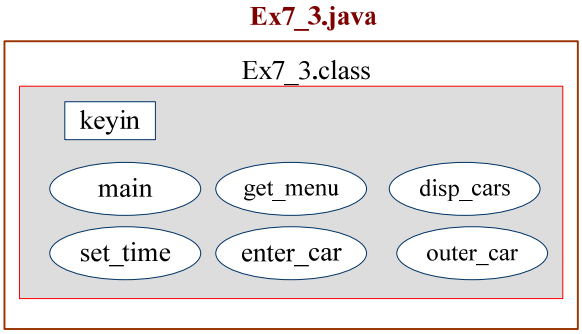

圖 7-16 Ex7_3 程式架構

```
01 //Ex7_3.java
02
```

```
03 /* 請建立一套停車場管理系統，功能有:顯示目前停車車輛、
04
05     車輛進入取票、車輛出場繳費 */
06
07 import java.io.*;
08 import java.util.*;
09 public class Ex7_3 {
10       static Scanner keyin = new Scanner(System.in);
11       public static void main(String args[]) {
12             Cars.number = 0;
13             int select = get_menu();
14             while (select != 5) {
15                   switch(select) {
16                         case 1:          // 顯示目前停車
17                               disp_cars();
18                               break;
19                         case 2:          // 設定現在時間
20                               set_time();
21                               break;
22                         case 3:          // 車輛進入
23                               enter_car();
24                               break;
25                         case 4:               // 車輛出場
26                               outer_car();
27                               break;
28                         default:
29                               System.out.printf("錯誤選擇, 請重新輸入\n");
30                   }
31                   select = get_menu();
32             }
33       }
34
35       /* 主功能表選單 */
36       static int get_menu()   {
37             System.out.printf("\n** 高雄市前金立體停車場 管理系統 目前時間");
38             System.out.printf(" %d 時 %d 分 **\n", Clock.hour, Clock.minute);
39             System.out.printf("(1) 顯示目前停車          (2) 設定目前時間\n");
40             System.out.printf("(3) 車輛進入取票          (4) 車輛出場繳費\n");
41             System.out.printf("(5) 離開系統\n");
42             System.out.printf("目前車輛 %d 請輸入工作選項 =>", Cars.number);
43             int select = keyin.nextInt();
44
45
46
47
48
```

```
49          keyin.nextLine();
50          return select;
51      }
52
53      /* 顯示目前停車車輛 */
54
55      static void disp_cars() {
56          System.out.printf("停車票號\t\t 進入時間\n");
57          for(int i=0; i<Cars.number; i++) {
58              System.out.printf("%d\t\t", Cars.car[i].getTicket());
59              System.out.printf("%d 時 %d 分\n", Cars.car[i].enterTime[0],
60                                  Cars.car[i].enterTime[1]);
61          }
62      }
63
64      /* 設定目前公用時間 */
65
66      static void set_time() {
67          System.out.printf("請輸入目前時間(時/分) =>");
68          String now = keyin.nextLine();
69          Scanner s = new Scanner(now).useDelimiter("/");
70          Clock.hour = s.nextInt();
71          Clock.minute = s.nextInt();
72      }
73
74      /* 車輛進場 */
75
76      static void enter_car() {
77          Cars.car[Cars.number] = new carTicket();
78          Cars.car[Cars.number].setTicket();
79          Cars.car[Cars.number].enterTime[0] = Clock.hour;       //記錄時
80          Cars.car[Cars.number].enterTime[1] = Clock.minute;      // 記錄分
81          int ticket = Cars.car[Cars.number].getTicket();
82          System.out.printf("請按<Enter>鍵取票 (票號：%d) =>", ticket);
83          keyin.nextLine();
84          Cars.number = Cars.number + 1;                          // 下一輛號碼
85       }
86
87       /* 車輛出場, 繳費 */
88       static void outer_car() {
89          System.out.printf("請出示停車票 =>");
90          int ticket = keyin.nextInt();
91          keyin.nextLine();
92          int flag =0, i;
93
94
```

```
95              for(i=0; i<Cars.number; i++) {                  // 搜尋該停車號
96                  if(ticket == Cars.car[i].getTicket()){
97                      flag = 1;
98                      break;
99                  }
100             }
101             if (flag == 0) {
102                 System.out.printf("查無此車輛 \n");
103                 return;
104             }
105             int hour = Clock.hour - Cars.car[i].enterTime[0];
106             int minute = Clock.minute - Cars.car[i].enterTime[1];
107             minute = minute + hour * 60;
108             int tax = (minute/30) * 20;
                for(int j=i; j<Cars.number; j++)            // 刪除該車票紀錄
                    Cars.car[j] = Cars.car[j+1];
                Cars.number = Cars.number - 1;              // 停車數 - 1
                System.out.printf("請繳交 %d 停車費(按<Enter>鍵) =>", tax);
                keyin.nextLine();
        }
    }
```

7-5 專題製作 – 儲蓄存款系統

『藝術銀行』期望製作一套『活期儲蓄存款系統』，程式設計師很難了解系統需求，因此依照可能狀態分段製作，每一階段完成並得到上級同意後，再往下一階段實現，吾人將各步驟分批實施如下。

7-5-1 範例研討：步驟(1)建立存款帳戶規格

(A) 系統功能：PM7_2_1.java、account.java

系統需要建立客戶帳戶。每一存款帳戶包含：姓名（String name）、帳戶（String No）與存款餘額（int balance），各項資料限制如下：

(1) 帳戶：由 12 個數字與 1 個檢查碼所構成（合計 13 碼），檢查碼計算方式是利用一個加權系列： 1 3 1 3 1 3...等 交替變換，帳號第一個數字乘以 1、第二位乘以 3、

第三位乘以 1...等依此類推，再將所乘的結果相加 (= total)，再取 10 的餘數 (value = total % 10)，檢查碼即是 10 減該餘數 (= 10 - value)。

(2) 存款餘額：如果取款後，餘額會少於 0，則會拒絕提款。

請製作一個帳戶類別 (Account.class) 是其具有上述功能，再至作主程式引用該類別，驗證是否正常。當建立帳戶時，只要輸入 12 個帳號，系統自動產生檢查碼；處理帳戶而輸入帳號，系統也會檢查檢查碼是否正確。

期望驗證帳戶類別功能的結果如下：

```
D:\Java2_book\chap7\PM7_2\PM7_2_1>javac Account.java        【編譯 Account.java】
D:\Java2_book\chap7\PM7_2\PM7_2_1>javac PM7_2_1.java        【編譯 PM7_2_1.java】
D:\Java2_book\chap7\PM7_2\PM7_2_1>dir/b
Account.class                                               【帳戶 Account 類別】
Account.java
PM7_2_1.class                                           【主程式 PM7_2_1 類別】
PM7_2_1.java

D:\Java2_book\chap7\PM7_2\PM7_2_1>java PM7_2_1               【執行主類別】
***建立新帳戶***                                             // 輸入帳戶驗證
請輸入姓名=>粘添壽
請輸入帳號(12 位數字) =>123451234589
****建立完成****
完整的新帳戶(13 碼)為=>12345123458910
請輸入存款金額 =>50000
餘額 = 50000
列印帳戶資料                                          // 輸出驗證結果
帳戶姓名： 粘添壽
帳戶號碼(13 碼)：12345123458910
餘額= 50000
```

(B) 帳戶類別(Account.class)：

首先必須建立帳戶類別，其內容包含帳戶姓名(name)、帳號(ID)與餘額(balance)，其中 ID

與 balance 為私有變數，必須透過物件方法存取，其架構如圖 7-11 所示。

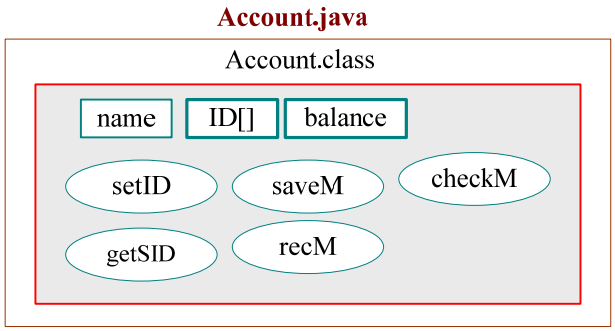

圖 7-17 Account.java 程式架構

```
01  //Account.java
02  /* 設定客戶姓名、帳號(ID) 與存款餘額 */
03  import java.io.*;
04  import java.util.Scanner;
05
06  class Account{
07        String name;
08        private int[]ID = new int[13];
09        private int balance;
10  // 設定帳號
11        int setID(String ID_Str) {
12              if(ID_Str.length() !=12){
13                    System.out.printf("需 12 個字元,請重新輸入!!\n");
14                    return 0;
15              }
16              else{
17                    Scanner s = new Scanner(ID_Str).useDelimiter("");
18                    int total = 0;
19                    for(int i=0;i<12;i++){
20                        ID[i]=s.nextInt();
21                        if((i+1)%2==0)
22                              total+=ID[i];
23                        else
24                              total+=ID[i]*2;
25                    }
26                    ID[12] = (10-(total%10));
27                    System.out.print("****建立完成****\n 完整的新帳戶(13 碼)為=>");
28                    for(int i=0;i<13;i++)
29                          System.out.print(ID[i]);
30                     System.out.println();
31                     return 1;
32
33
```

```
34      }
35       }
36
37  // 取得帳號
38      int[] getID(){
39          return ID;
40      }
41
42  // 寫入帳號
43      int writeID(String ID_STR1) {
44          if(ID_STR1.length() !=13){
45              System.out.printf("長度不對\n");
46              return 0;
47          }
48          else{
49              Scanner s = new Scanner(ID_STR1).useDelimiter("");
50              int total = 0, check, check_R;
51              for(int i=0;i<12;i++){
52                  ID[i]=s.nextInt();
53                  if((i+1)%2==0)
54                      total+=ID[i];
55                  else
56                      total+=ID[i]*2;
57              }
58              check_R = s.nextInt();
59              check = 10-(total%10);
60              if (check == check_R) {
                    ID[12] = check;
                    return 1;
                } else {
                    System.out.printf("檢查碼不對\n");
                    return 0;
                }
            }
        }

    // 存款
        int saveM(int money){
            balance = balance + money;
            return balance;
        }
    // 取款
        int recM(int money) {
            int m = balance - money;
```

```
            if (m >=0) {
                balance = m;
                return balance;
            }
            else {
                return -1;
            }
        }
    // 查詢餘額
        int checkM(){
            return balance;
        }
    }
```

(C) 主類別(PM7_2_1.class)：

我們寫一個簡單程式來驗證 Account 類別是否能滿足所需，程式範例如下：

```
01  //PM7_2_1.java
02
03  import java.util.*;
04  public class PM7_2_1{
05      public static void main(String[] args) {
06          Scanner keyin = new Scanner(System.in);
07          String ID_Str;
08          Account customer = new Account();
09          System.out.printf("***建立新帳戶***\n 請輸入姓名=>");
10          customer.name = keyin.nextLine();
11          System.out.printf("請輸入帳號(12 位數字) =>");
12          ID_Str = keyin.nextLine();
13          int flag = customer.setID(ID_Str);
14          if (flag == 0)
15              return;
16          System.out.printf("請輸入存款金額 =>");
17          int money = keyin.nextInt();
18          int balance = customer.saveM(money);
19          System.out.printf("餘額 = %d\n", balance);
20  /* 列印帳戶清單 */
21          System.out.printf("列印帳戶資料\n");
22          System.out.printf("帳戶姓名： %s \n", customer.name);
23          int[] ID = customer.getID();
24          System.out.printf("帳戶號碼(13 碼)：");
25
26
27
28
```

```
29              for (int i=0; i<13; i++)
30                  System.out.printf("%d", ID[i]);
31              System.out.printf("\n");
                System.out.printf("餘額= %d\n", customer.checkM());
            }
        }
```

7-5-2 範例研討：步驟(2)建立儲蓄存款系統

(A) 系統功能：PM7_2_2.java、Account.java

帳戶規格 (Account.class) 經過驗證可行之後，期望您進一步請幫『藝術銀行』建立一套『儲蓄存款系統』的雛形系統，包含有下列功能：

(1) 讀入現有帳戶資料(deposit.data)。

(2) 顯示帳戶：顯示系統每一帳戶的姓名、帳號與餘額。

(3) 存款：客戶提款功能，選擇後會要求輸入客戶帳號，再要求輸入存款金額，執行後會顯示目前餘額。

(4) 提款：選擇後會要求輸入客戶帳號，再要求輸入提款金額，執行後會顯示目前餘額。

(5) 查詢餘額：選擇後會要求輸入客戶帳號，再顯示餘額。

(6) 儲存資料：將資料寫入 deposit.data 檔案內。

沿用設計完成的 Account.class 類別，並利用它產生一個物件陣列，儲存客戶帳戶資料。系統啟動時，必須將 deposit.data（必須事先產生）資料寫入陣列內。

(1) 編譯後，有下列檔案：

```
D:\Java2_book\chap7\PM7_2_2>dir/b
Account.class                   【帳戶類別】
Account.java
deposit.data                    【資料檔案】
PM7_2_2.class                   【主類別】
PM7_2_2.java
```

(2) 主選單有 8 個選項如下：

```
D:\Java2_book\chap7\PM7_2_2>java PM7_2_2
== 藝術銀行 活期儲蓄系統 ==
(1) 讀入所有帳戶        (2) 建 立 新帳戶        (3) 顯示所有帳戶
(4) 帳戶提款作業        (5) 帳戶存款作業        (6) 帳戶查詢餘額
(7) 儲 存 資 料         (8) 離 開 系 統
請選擇工作項目 =>
```

(3) 首先由檔案讀入資料(選項 1)，再觀察目前所有帳戶的資料(選項 2)，如下：

```
請選擇工作項目 => 1

= 藝術銀行 活期儲蓄系統 ==
1) 讀入所有帳戶        (2) 建 立 新帳戶        (3) 顯示所有帳戶
4) 帳戶提款作業        (5) 帳戶存款作業        (6) 帳戶查詢餘額
7) 儲 存 資 料         (8) 離 開 系 統
請選擇工作項目 => 3
= 列印所有帳戶資料 ==
帳號              姓名     餘額
234567812344    粘添壽   60000
345678901238    劉聲聲   60000
```

(4) 建立新帳戶(選項 2) 操作如下：

```
請選擇工作項目 => 2
**建立新帳戶***
請輸入姓名=>張美人
請輸入帳號(12 位數字) =>345678901234
***建立完成****
完整的新帳戶(13 碼)為=>34567890123410
請輸入存款金額 =>50000
餘額 = 50000
```

(5) 客戶提款(選項 4) 的操作如下：

```
請選擇工作項目 => 4
請輸入存款帳號 =>1234567812344
請輸入提款金額 =>6000
帳戶：粘添壽目前餘額為：54000
```

(6) 客戶存款作業(選項 5)如下：

```
請選擇工作項目 => 5
請輸入提款帳號 =>1234567812344
請輸入存款金額 =>70000
帳戶：粘添壽目前餘額為：124000
```

(7) 客戶查詢餘額(選項 6) 的操作如下：

```
請選擇工作項目 => 6
請輸入存款帳號 =>1234567812344
帳戶：粘添壽目前餘額為：124000
```

(8) 儲存檔案(選項 7) 的操作如下：

```
請選擇工作項目 => 7
** 將儲存檔案 (depoist.dada) **
***** 儲存完畢 ****
```

(B) 系統分析

請讀者自行研討。

(C) 程式範例

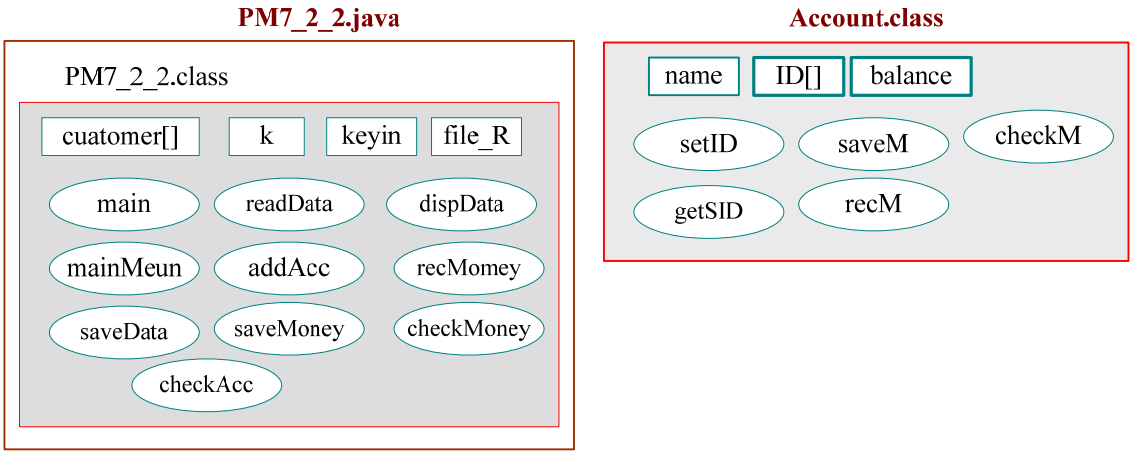

圖 7-18 PM7_2_1 程式架構

```
01  //PM7_2_2.java
02
03  import java.io.*;
04  import java.util.*;
05  public class PM7_2_2 {
06        static Account[] customer;        // 客戶資料的物件陣列
07
08        static int k=0;                          // 紀錄儲存筆數
09
09        static Scanner keyin;     // 鍵盤輸入物件
10
10        static String file_R = "deposit.data";
11
12        public static void main(String args[]) throws IOException {
13            keyin = new Scanner(System.in);
14            customer = new Account[50];
15
16            mainMenu();
17            int select = keyin.nextInt();
18            keyin.nextLine();
19            while (select !=8) {
20                  switch (select) {
21
22                        case 1:         /* 讀入帳戶資料 */
23                              readData();
24                              break;
25                        case 2:         /* 新建立帳戶 */
26                              addAcc();
27                              break;
28
29                        case 3:         /* 顯示所有帳戶 */
30                              dispData();
31                              break;
32                        case 4:         /* 提款作業 */
```

```
33                          recMoney();
34                          break;
35                  case 5:             /* 存款作業 */
36                          saveMoney();
37                          break;
38                  case 6:             /* 查詢餘額 */
39                          checkMoney();
40                          break;
41                  case 7:             /* 儲存帳戶資料 */
42                          saveData();
43                          break;
44                  default:
45                          System.out.printf("錯誤輸入, 請重新選擇 !!\n");
46              }
47              mainMenu();
48              select = keyin.nextInt();
49              keyin.nextLine();
50          }
51      }
52      /* 顯示主選單 */
53      public static void mainMenu() {
54          System.out.printf("\n== 藝術銀行 活期儲蓄系統 ==\n");
55          System.out.printf("(1) 讀入所有帳戶\t");
56          System.out.printf("(2) 建 立 新帳戶\t");
57          System.out.printf("(3) 顯示所有帳戶\n");
58          System.out.printf("(4) 帳戶提款作業\t");
59          System.out.printf("(5) 帳戶存款作業\t");
60          System.out.printf("(6) 帳戶查詢餘額\n");
61          System.out.printf("(7) 儲 存 資 料\t");
62          System.out.printf("(8) 離 開 系 統\n");
63          System.out.printf("請選擇工作項目 => ");
64      }
65      /* 讀取 deposit.data 檔案資料, 再存入 customer 物件陣列中 */
66      public static void readData() throws IOException {
67          String inData;
68          String ID_STR1;
69          int money;
70          File fileID = new File(file_R);      // 產生輸入檔案物件
71
72
73
74
75
76
77
78
```

```
79              if (fileID.exists()) {
80                    BufferedReader data = new BufferedReader(new
81                                                          FileReader(fileID));
82                    while ((inData=data.readLine()) != null) {
83                          Scanner s = new Scanner(inData).useDelimiter("\t");
84                          customer[k] = new Account();
85                          ID_STR1 = s.next();
86                          if (customer[k].writeID(ID_STR1) == 0) {
87                                System.out.printf("%s 帳戶不對\n", ID_STR1);
88                                continue;
89                          }
90                          customer[k].name = s.next();
91                          money = s.nextInt();
92                          customer[k].saveM(money);
93                          k = k +1;
94                    }
95                    data.close();
96              }
97              else {
98                    System.out.printf("%s 檔案不存在, 請先建立它\n", file_R);
99                    System.out.printf("按任何鍵離開 =>");
100                   keyin.nextLine();
101                   System.exit(1);
102             }
103       }
104       public static void addAcc() {
105             String ID_Str;
106             customer[k] = new Account();
107             System.out.printf("***建立新帳戶***\n 請輸入姓名=>");
108             customer[k].name = keyin.nextLine();
109             System.out.printf("請輸入帳號(12 位數字) =>");
110             ID_Str = keyin.nextLine();
111             int flag = customer[k].setID(ID_Str);
112             if (flag == 0)
113                   return;
114             System.out.printf("請輸入存款金額 =>");
115             int money = keyin.nextInt();
116             int balance = customer[k].saveM(money);
117             System.out.printf("餘額 = %d\n", balance);
118             k = k + 1;
119
120       }
121
122
123
124
```

```
125     /* 顯示所有帳戶 */
126     public static void dispData() {
127         System.out.printf("== 列印所有帳戶資料 ==\n");
128         System.out.printf("帳號\t\t 姓名\t 餘額\n");
129         for (int i=0; i<k; i++) {
130             int[] ID = customer[i].getID();
131             for (int j=0; j<13; j++)
132                 System.out.printf("%d", ID[j]);
133             System.out.printf("\t");
134             System.out.printf("%s\t", customer[i].name);
135             System.out.printf("%d\n", customer[i].checkM());
136         }
137     }
138     /* 檢查帳號 */
139     public static int checkAcc(String ID_Str) {
140         int[] ID = new int[13];
141         int[] ID_R = new int[13];
142         int acc, flag=0;
143         Scanner s = new Scanner(ID_Str).useDelimiter("");
144         for (int i=0; i<13; i++)
145             ID[i] = s.nextInt();
146         for (acc=0; acc<k; acc++) {
147             ID_R = customer[acc].getID();
148             if (Arrays.equals(ID, ID_R)) {
149                 flag =1;
150                 break;
151             }
152         }
153          if (flag == 0) {
154             System.out.printf("無此帳號，請離開 !!\n");
155             return -1;
156          }else {
157             return acc;
158          }
159     }
160
161     /* 存款作業 */
162     public static void saveMoney() {
163         int[] ID = new int[13];
164         int flag;
165         System.out.printf("請輸入提款帳號 =>");
166         String ID_Str = keyin.nextLine();
167         int acc = checkAcc(ID_Str);
168
169
170
```

```
171             if (acc < 0) {
172                 System.out.printf("無此帳號，請離開 !!\n");
173                 return;
174             }
175             System.out.printf("請輸入存款金額 =>");
176             int money = keyin.nextInt();
177             int m = customer[acc].saveM(money);
178             String name = customer[acc].name;
179             System.out.printf("帳戶：%s 目前餘額為：%d\n", name, m);
180         }
181         /* 提款作業 */
182         public static void recMoney(){
183             System.out.printf("請輸入存款帳號 =>");
184             String ID_Str = keyin.nextLine();
185             int acc = checkAcc(ID_Str);
186             if (acc < 0) {
187                 System.out.printf("無此帳號，請離開 !!\n");
188                 return;
189             }
190             System.out.printf("請輸入提款金額 =>");
191             int money = keyin.nextInt();
192             int m = customer[acc].recM(money);
193             if(m <0){
194                 System.out.printf("存款不足、請離開 !!\n");
195                 return;
196             }
197             String name = customer[acc].name;
198             System.out.printf("帳戶：%s 目前餘額為：%d\n", name, m);
199         }
200        /* 查詢餘額 */
201         public static void checkMoney(){
202             System.out.printf("請輸入存款帳號 =>");
203             String ID_Str = keyin.nextLine();
204             int acc = checkAcc(ID_Str);
205             if (acc < 0) {
206                 System.out.printf("無此帳號，請離開 !!\n");
207                 return;
208             }
209             int m = customer[acc].checkM();
210             String name = customer[acc].name;
```

```
217                 System.out.printf("帳戶：%s 目前餘額為：%d\n", name, m);
218             }
219             /* 儲存資料 */
220             public static void saveData() throws IOException {
221                 BufferedWriter outData = new BufferedWriter(new
222                                                         FileWriter(file_R));
223                 int[] ID_R = new int[13];
224                 System.out.printf("** 將儲存檔案 (depoist.dada) **\n");
225                 for (int i=0; i<k; i++) {
226                     long temp=0;
                        ID_R = customer[i].getID();
                        for(int j=0; j<13; j++) {
                            temp = temp + ID_R[j];
                            temp = temp * 10;
                        }
                        temp = temp /10;
                        outData.write(temp + "\t");
                        outData.write(customer[i].name + "\t");
                        outData.write(customer[i].checkM() + "\t");
                        outData.newLine();
                    }
                    outData.close();
                    System.out.printf("***** 儲存完畢 ****\n");
                }
            }
```

7-5-3 自我挑戰：步驟(3)建立交易檔規格

(A)系統功能：PM7_2_3.java、Account.java、transcation.java

為了安全起見，一般銀行大多會儲存客戶處理帳戶的過程，可作為客戶查詢交易過程使用。因此，『藝術銀行』期望建立一個交易檔（Transcation.data）紀錄每一帳戶的每一筆交易（提款或存款），交易檔的每一筆資料記錄有：帳戶（String No）、日期（String date）、交易種類（int type，1 表存款；0 表提款）、交易量（int trade），其中交易量正數表示存款；負數表示提款。請製作一個具有上述功能的類別（Transcation.class），並利用主程式輸入各項資料後，印出資料觀察是否可行。

7-5-4 自我挑戰：步驟(4)建立安全性存款系統

(A)系統功能：PM7_2_4.java、Account.java、transcation.java

交易檔類別建立之後，接著由 PM7_2_2.java 擴充。當客戶處理存款或提款功能時，皆會將交易資料寫入利用 Transcation.class 所建立的物件陣列內。功能選項也增加下列功能：

(1) 『顯示交易資料』: 顯示所有客戶的交易資料。

(2) 『儲存交易資料』: 將帳戶資料寫入 deposit.data 檔案，也將交易資料寫入 transaction.data 檔案內。

(3) 『客戶交易查詢』: 可供以帳號，查詢帳戶所有交易資料。