CH10 上課練習

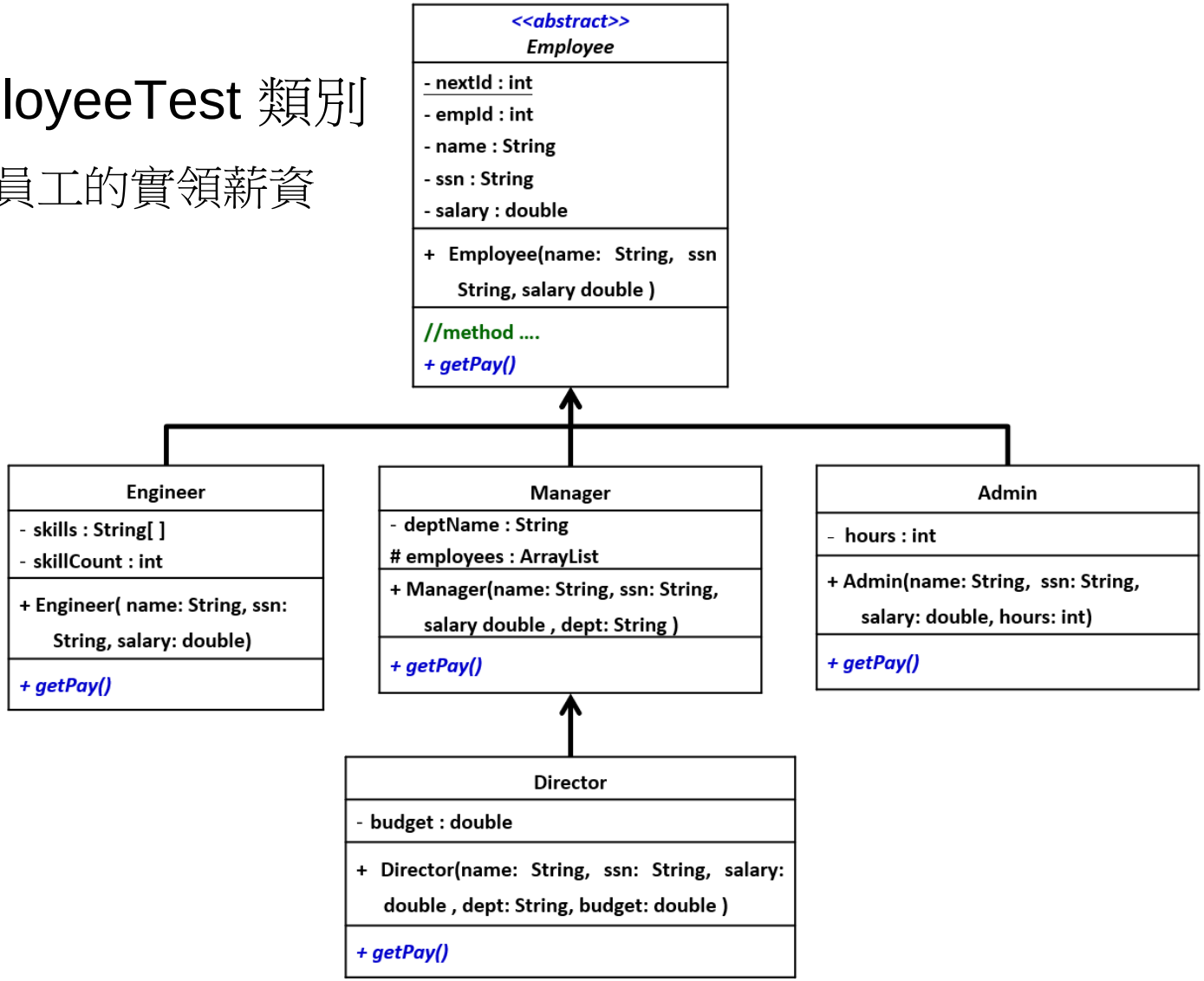
鄭安翔
ansel_cheng@hotmail.com

練習一

1. 修改 EmployeePractice 專案
2. 修改 Employee 類別為抽象類別
   - 宣告 getPay( )抽象方法
3. 修改 Admin 類別
   - 新增工時的屬性並修改建構子
   - 覆寫 getPay( ) ，金額為月薪乘工作時數除以160(20工作天*8小時)
4. 修改 Engineer 類別
   - 覆寫 getPay( ) ，每一項技能可領3000加給
5. 修改 Manager, 類別
   - 覆寫 getPay( ) ，管理一個員工可領2000元加給
6. 修改 Director 類別
   - 覆寫 getPay( ) ，管理一個經理可領10000元加給

練習一

7. 修改 EmployeeTest 類別
   - 列印所有員工的實領薪資



Employee 類別

*Employee.java

```java
package com.example.domain;

import java.text.NumberFormat;

public abstract class Employee {
    public static int nextId = 101;
    private int empId;
    private String name = "John";
    private String ssn = "A123456789";
    private double salary = 26400;
    protected NumberFormat formatter = NumberFormat.getCurrencyInstance();

    public Employee(String name, String ssn, double salary) {...

    public abstract double getPay();

    public int getEmpId() {...

    public String getName() {...

    public void setName(String name) {...

    public String getSsn() {...

    public double getSalary() {...

    public void raiseSalary(double increase) {...

    public String toString() {...

    public int hashCode() {...

    public boolean equals(Object obj) {...

}
```

Admin 類別

Admin.java

```java
package com.example.domain;

public class Admin extends Employee {
    private int hours;

    public Admin(String name, String ssn, double salary, int hours) {
        super(name, ssn, salary);
        this.hours = hours;
    }

    @Override
    public double getPay() {
        return this.getSalary()/160*hours;
    }

}
```

Engineer 類別

Engineer.java

```java
package com.example.domain;

public class Engineer extends Employee {
    private String[] skills;
    private int skillCount;

    public Engineer(String name, String ssn, double salary) {
        super(name, ssn, salary);
        skills = new String[5];
        skillCount = 0;
    }

    public void addSkill(String skill) {
        if(skillCount<5)
            skills[skillCount++] = skill;
        else
            System.out.println("最多註冊五種技能,新增失敗!");
    }

    @Override
    public double getPay() {
        return this.getSalary() + skillCount*3000;
    }

    public String toString() {

}
```

Manager 類別

Manager.java

```java
package com.example.domain;

import java.util.ArrayList;

public class Manager extends Employee {
    private String deptName;
    protected ArrayList employees;

    public Manager(String name, String ssn, double salary, String deptName) {
        super(name, ssn, salary);
        this.deptName = deptName;
        this.employees = new ArrayList();
    }

    @Override
    public double getPay() {
        return this.getSalary()+employees.size()*2000;
    }

    public String getDeptName() {...

    public boolean addEmployee(Employee emp) {...

    public boolean removeEmployee(Employee emp) {...

    public String getStaffDetails() {...

    public String toString() {...

}
```

Director 類別

Director.java

```java
package com.example.domain;

public class Director extends Manager {
    private double budget;

    public Director(String name, String ssn, double salary, String deptName, double budget) {
        super(name, ssn, salary, deptName);
        this.budget = budget;
    }

    @Override
    public double getPay() {
        return this.getSalary()+employees.size()*10000;
    }

    public double getBudget() {...

    public String toString() {...

}
```

EmployeeTest 類別

EmployeeTest.java

```java
package com.example;

import com.example.domain.Admin;

public class EmployeeTest {

    public static void main(String[] args) {
        Employee[] emps = new Employee[5];
        emps[0] = new Admin("Sean", "A123456789", 50000, 180);
        emps[1] = new Admin("Amy", "B210987654", 70000, 120);
        emps[2] = new Engineer("David", "C109876543", 80000);
        emps[3] = new Manager("Louis", "D124680135", 100000, "TW Sales");
        emps[4]= new Director("Nicole", "R202468135", 120000, "Global Sales", 1000000);

        for(int i=0; i<emps.length; i++)
            System.out.println(emps[i]);

        System.out.println("David 學會了Java, Android");
        if(emps[2] instanceof Engineer) {
            Engineer eng = (Engineer )emps[2];
            eng.addSkill("Java");
            eng.addSkill("Android");
        }

        System.out.println("部門分配.....");
        if(emps[3] instanceof Manager) {
            Manager m1 = (Manager)emps[3];
            m1.addEmployee(emps[0]);
            m1.addEmployee(emps[1]);
            m1.addEmployee(emps[2]);
        }

        ((Manager)emps[4]).addEmployee(emps[3]);

        for(int i=0; i<emps.length; i++)
            System.out.println(emps[i].getName()+"本月薪資"+emps[i].getPay()+"元");

    }
}
```

測試

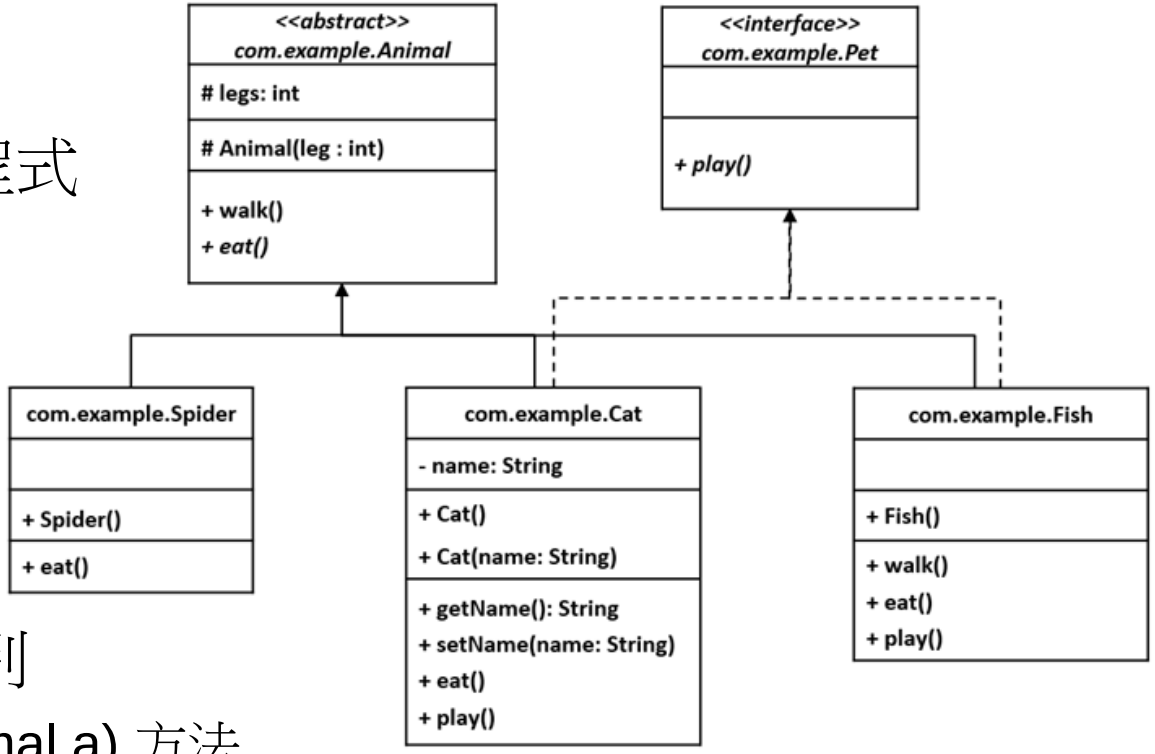
Console

```
<terminated> EmployeeTest [Jav
======員工資料======
編號: 101
姓名: Sean
SSN: A123456789
薪水: $50,000.00元

======員工資料======
編號: 102
姓名: Amy
SSN: B210987654
薪水: $70,000.00元

======員工資料======
編號: 103
姓名: David
SSN: C109876543
薪水: $80,000.00元

======員工資料======
編號: 104
姓名: Louis
SSN: D124680135
薪水: $100,000.00元
管理部門: TW Sales

======員工資料======
編號: 105
姓名: Nicole
SSN: R202468135
薪水: $120,000.00元
管理部門: Global Sales
管理預算: $1,000,000.00
```

```
David 學會了Java, Android
部門分配.....
Sean本月薪資56250.0元
Amy本月薪資52500.0元
David本月薪資86000.0元
Louis本月薪資106000.0元
Nicole本月薪資130000.0元
```

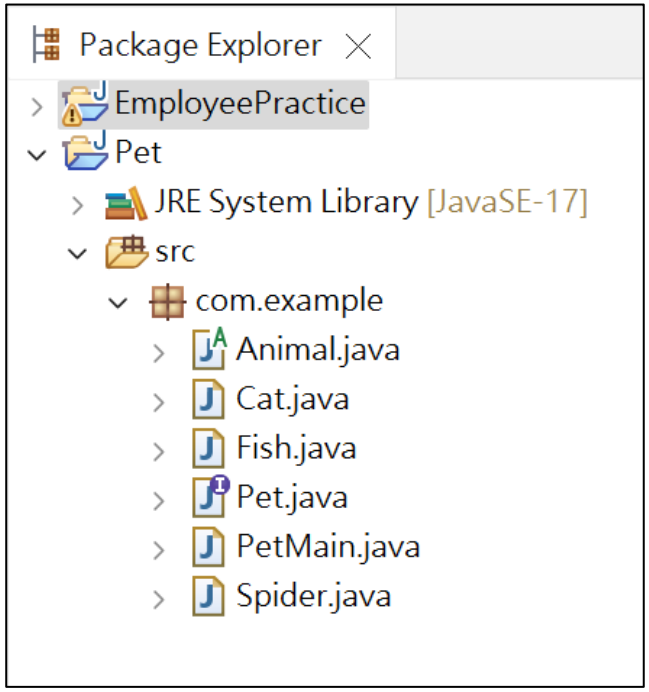
練習二 Interface

- 開啟 Pet 專案
- 依如右圖設計撰寫程式
  - 抽象類別 Animal
  - 介面 Pet
  - Spider 類別
  - Cat 類別
  - Fish 類別
- 撰寫 PetMain 主類別
  - playWithAnimal(Animal a) 方法
    - 寵物, 呼叫 play( ) 方法
    - 非寵物輸出警告訊息
  - 建立Animal 陣列
    - 置入Spider、Cat 、Fish 物件



  - 測試 walk( ) 及 eat( ) 方法
  - 測試 playWithAnimal( ) 方法

開啟 Pet 專案


Package Explorer
EmployeePractice
Pet
JRE System Library [JavaSE-17]
src
com.example
Animal.java
Cat.java
Fish.java
Pet.java
PetMain.java
Spider.java

Animal 類別

Animal.java

```java
package com.example;

public abstract class Animal {
    protected int legs;

    protected Animal(int legs) {
        this.legs = legs;
    }

    public void walk() {
        System.out.printf("用%d隻腳走路%n", legs);
    }

    public abstract void eat();

}
```

Pet.java介面

Pet.java

```java
package com.example;

public interface Pet {

    public void play();

}

```

Cat.java類別

Cat.java

```java
package com.example;

public class Cat extends Animal implements Pet{

    private String name;


    public Cat() {
        super(4);
        this.name="";
    }

    public Cat(String name) {
        super(4);
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public void eat() {
        if(name!=null && name.length()!=0)
            System.out.println(name+"最喜歡吃魚");
        else
            System.out.println("貓最喜歡吃魚");
    }

    @Override
    public void play() {
        System.out.printf("%s玩躲貓貓%n", name);
    }

}
```

Fish.java類別

Fish.java

```java
package com.example;

public class Fish extends Animal implements Pet{

    public Fish() {
        super(0);
    }

    @Override
    public void eat() {
        System.out.println("大魚吃小魚");
    }

    @Override
    public void walk() {
        System.out.println("魚沒有腳,只會游泳");
    }

    @Override
    public void play() {
        System.out.println("靜靜地欣賞!");
    }


}
```

Spider 類別

Spider.java

```java
package com.example;

public class Spider extends Animal {

    public Spider() {
        super(8);
    }

    @Override
    public void eat() {
        System.out.println("蜘蛛吃蚊子");
    }

}
```

主類別 PetMain

PetMain.java

```java
package com.example;

public class PetMain {

    public static void main(String[] args) {
        Animal[] animals = new Animal[3];

        animals[0] = new Spider();
        animals[1] = new Cat("加菲");
        animals[2] = new Fish();

        System.out.println("====測試eat()/walk()/play()====");
        for(Animal a : animals) {
            a.eat();
            a.walk();
            playWithAnimal(a);
            System.out.println();
        }

    }

    public static void playWithAnimal(Animal a) {
        if(a instanceof Pet)
            ((Pet)a).play();
        else
            System.out.println("危險,這不是寵物,不要靠近!");
    }

}
```

測試、執行

```
Console
<terminated> PetMain [Java Application] C:\
====測試eat()/walk()/play()====
蜘蛛吃蚊子
用8隻腳走路
危險,這不是寵物,不要靠近!

加菲最喜歡吃魚
用4隻腳走路
加菲玩躲貓貓

大魚吃小魚
魚沒有腳,只會游泳
靜靜地欣賞!
```