

第10章 面向对象思考

目录

contents



10.1 了解软件开发过程



10.2 掌握类间的关系描述方法



10.3 了解类的设计原则



10.4 设计实例

10.1 了解软件开发过程

- 面向过程开发：一个软件系统由一系列过程构成。因而采用功能划分或模块分解的方法进行。核心思想是将问题分解为多个小问题, 通过传递数据、解决这些小问题来解决整个问题。
- 面向对象开发：一种基于对象概念的系统开发方法，一个软件系统由一系列参与活动的对象构成。
 - ⑩ 系统调查和需求分析。对系统将要面临的管理问题以及用户对系统的功能需求进行调查研究。
 - ⑩ 面向对象的分析。抽象地识别出对象和对象的行为、结构、属性及方法等，进而确定类。
 - ⑩ 面向对象的设计。对上一阶段的结果整理、归类和修正，以规范的形式确定下来，包括设计数据库结构、确定系统结构、定义属性和方法等。
 - ⑩ 面向对象的编程。采用面向对象的设计语言将设计模型编为程序。

10.1 了解软件开发过程

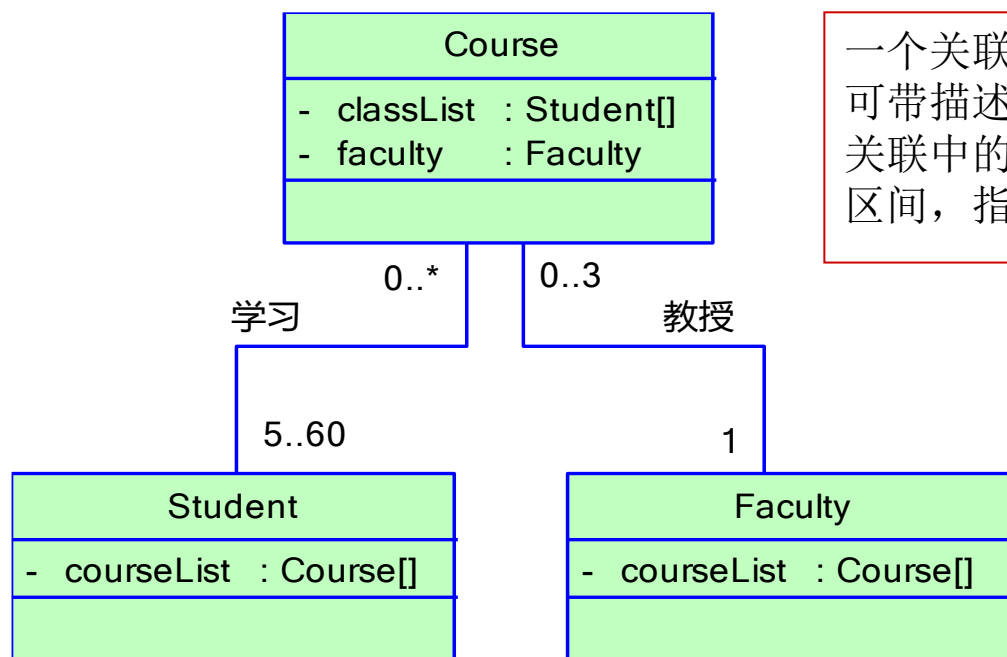
- 面向对象开发方法还包括系统实现、测试和维护等阶段。这些方法中UML（统一建模语言）是一种广泛使用的标准建模语言，它支持从需求分析开始的软件开发全过程，包括用例驱动、以体系结构为中心、迭代和渐增式的开发过程。

建立对象模型、动态模型和功能模型。

- 对象模型：描述对象的组织结构。是核心模型。
- 动态模型：描述对象之间的交互行为。
- 功能模型：描述对象的行为或状态变化。
- 对象模型：描述类和类之间的关系，包括：关联、聚合、组合、依赖、继承、实现。
- UML建模语言的类图能够描述对象（类）的组织结构和行为。

10.2 掌握类间的关系描述方法-关联关系

- 关联关系 (association) 是一种通用的二元关系，对象间通过活动发生联系。例如，学生(Student)选学课程(Course)，教师(Faculty)教授课程(Course)，这些联系可以在UML中表示。



一个关联可以用两个类之间的实线（可带双向箭头）表示。
可带描述关系的标签。
关联中的每个类可以指定一个数目或数字
区间，指出这个关系涉及多少个对象

10.2 掌握类间的关系描述方法-关联关系实现

- 在Java代码中，关联关系可以用数据域或方法来实现。
对于方法，一个类中的方法包含另一个类的参数。

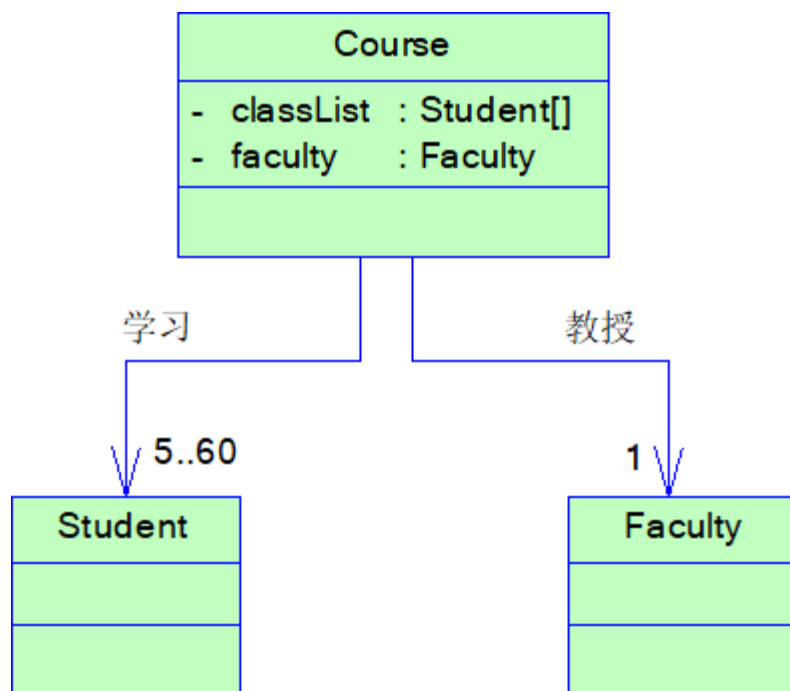
```
public class Student{  
    private Course[] courseList;//一对多用数组实现  
    public void addCourse( Course c){ ..... }  
}
```

```
public class Course{  
    private Student[] classList;  
    private Faculty faculty;  
    public void addStudent( Students s){ ..... }  
    public void setFaculty( Faculty f) { ..... }  
}
```

```
public class Faculty{  
    private Course[] courseList;  
    public void addCourse( Course c){ ..... }  
}
```

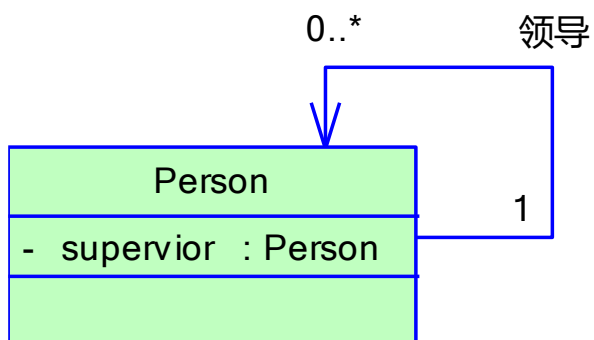
10.2 掌握类间的关系描述方法-单向关联

- 如果学生或教师不需要知道课程的信息，可以去掉courseList域，形成单向关联。这时可将Student类和Faculty类中的数据域courseList和addCourse方法去掉。



10.2 掌握类间的关系描述方法-自关联

- 同一个类的对象间存在关联，称自关联。例如，一个人有一个领导。



```
public class Person{  
    private Person supervisor;  
    .....  
}
```

10.2 掌握类间的关系描述方法-聚合和组合

- 聚合关系 (aggregation) 是一种拥有关系，表示整体与部分之间的关系，即 **has-a** 的关系。所有者成为聚集者，从属对象称为被聚集者。在聚合关系中，一个对象可以被多个聚集者拥有 (Weak has a)。
- 组合关系 (composition) 是一种隶属关系，表示从属者强烈依赖于聚集者。一个从属者只能被一个聚集者所拥有，聚集者负责从属者的创建和销毁 (Strong has a)。



一个Name对象只能为一个Person所有，但一个Address对象可以被多个Person共享

10.2 掌握类间的关系描述方法-聚合和组合

- 聚集关系和组合关系在代码中通常表示为聚集类中的数据域，如上图中的关系可以表示为

```
public class Name{  
    ...  
}
```

```
public class Person{  
    private Name name;  
    private Address address;  
    ...  
}
```

```
public class Address{  
    ...  
}
```

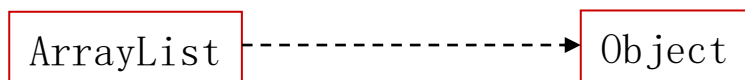
- 对于组合关系，聚集者往往负责对从属者的创建和销毁，而聚集关系则不是。

Address对象是传递进来的一个引用，而Name对象是在Person对象创建时才创建。
当Person对象被析构时，Name对象也被析构，而Address对象可能还存在

```
public class Person{  
    private Name name;  
    private Address address;  
    public Person(Address a){  
        name = new Name();  
        address = a;  
    }  
}
```

10.2 掌握类间的关系描述方法-依赖关系

- 依赖关系（dependency）指的是两个类之间一个（称为client）依存另一个（称为supplier）的关系。在UML中，从client画一条带箭头的虚线指向supplier类。
- 例如，可以向容器类ArrayList添加对象，因此ArrayList和Object之间的关系可以用依赖描述。

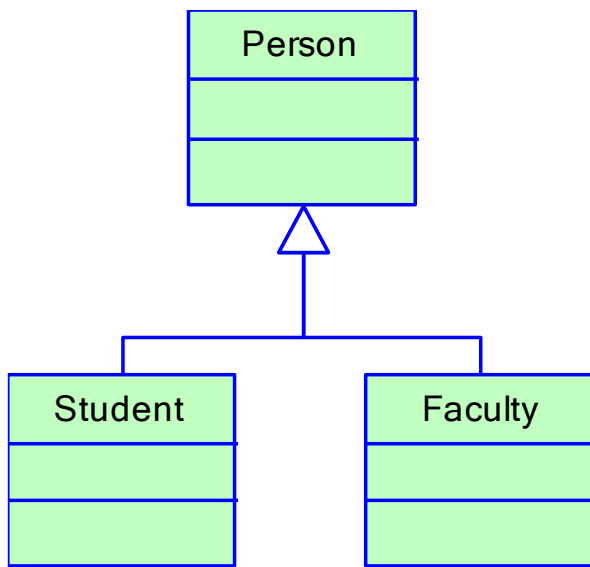


- 依赖关系在代码中通常表示为client类的成员函数以supplier类型的对象为参数

```
public class ArrayList{  
    public void add(Object o){  
        ...  
    }  
}
```

10.2 掌握类间的关系描述方法-继承关系

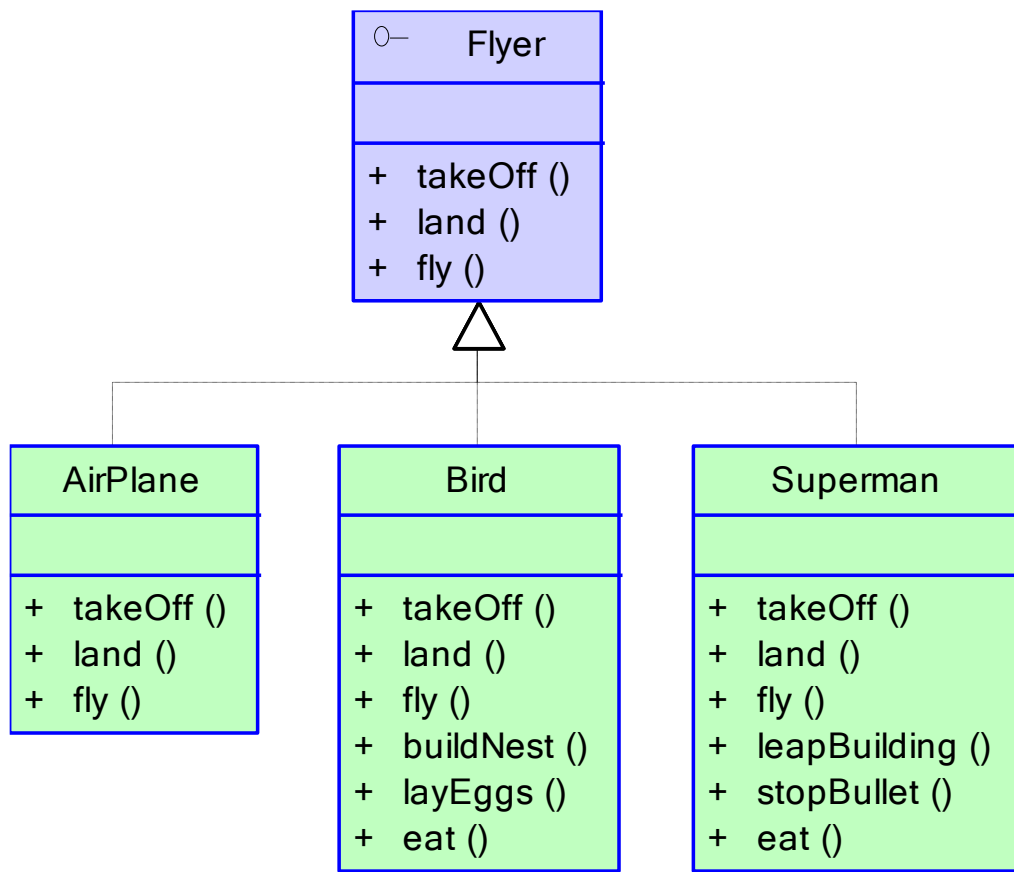
- 继承关系 (inheritance) 表示子类与父类之间的is-a关系。（泛化(Generalization)）



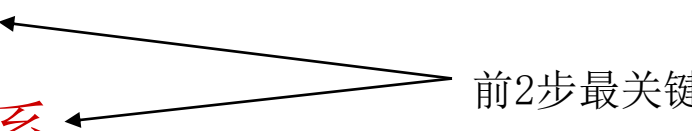
- 通过继承，子类可以重用父类的数据和代码。

10.2 掌握类间的关系描述方法-实现关系

□ 实现关系 (realization) 表示类和接口之间的关系。

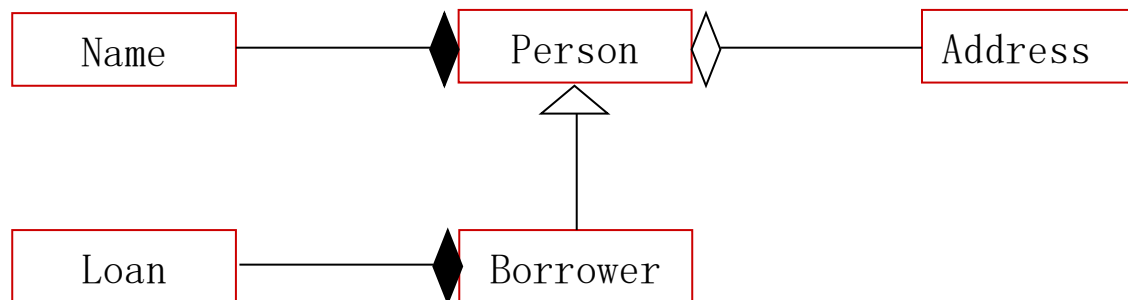


10.3 掌握类间的关系描述方法-类的设计原则

- 确定系统中的类
 - 建立类之间的关系
 - 描述每个类的属性和方法
 - 编写类的代码
 - 例如建立一个借方和贷款的模型，借方是要贷款的人，人有姓名和地址，因此可以确定以下类：
 - 人Person
 - 姓名Name
 - 地址Address
 - 借方Borrow
 - 一笔贷款Loan
- 
- 前2步最关键

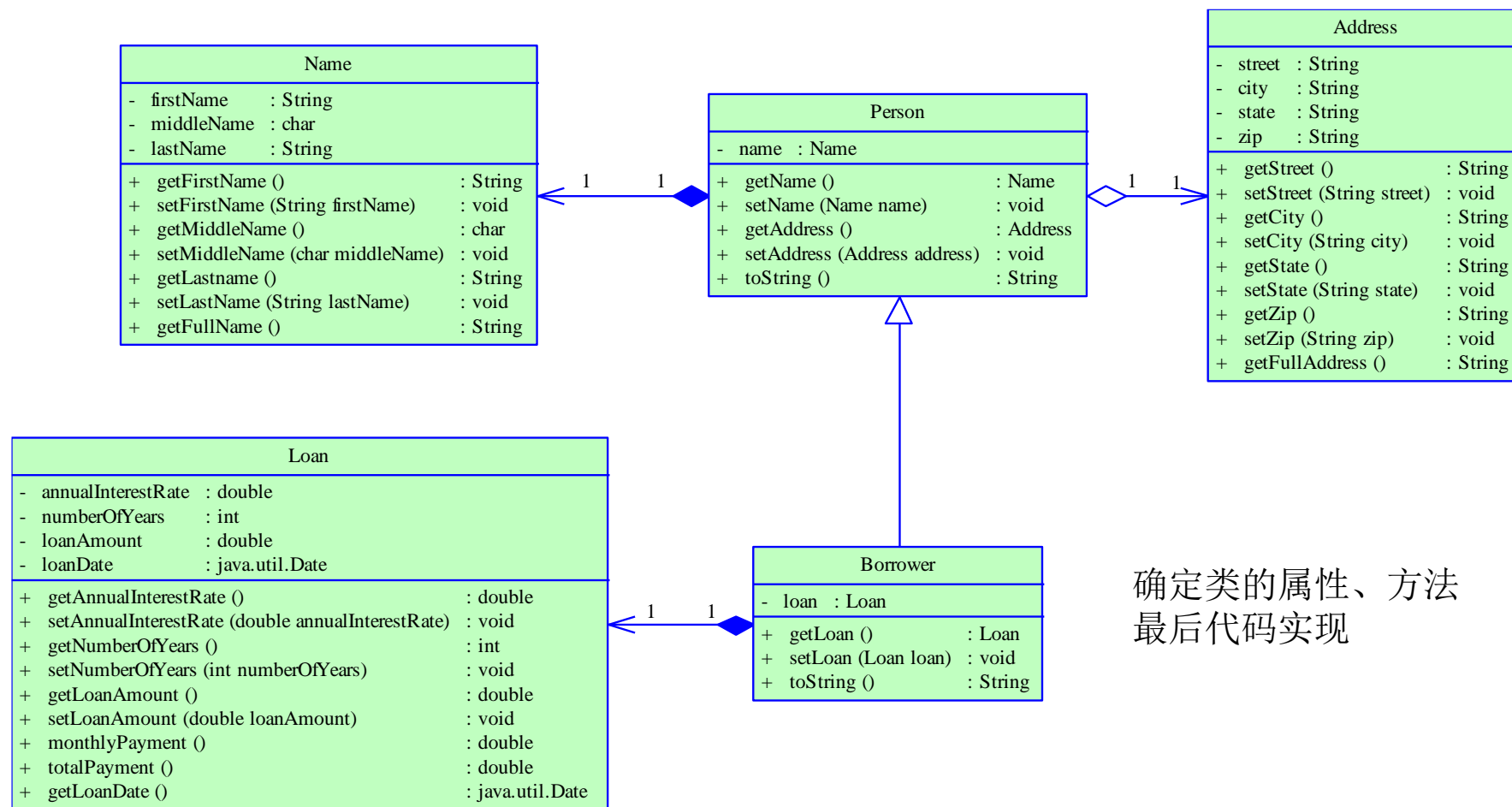
10.3 掌握类间的关系描述方法-类的设计原则

□ 分析类的关系



- Person对象包含Name对象和Address对象，Person和Name之间为组合关系，Person和Address之间为聚集关系
- Borrower继承Person
- Borrower对象包含一笔贷款，Borrower和Loan之间为组合关系。

10.4 设计实例-贷款的类型模型



确定类的属性、方法
最后代码实现