C++ Programming

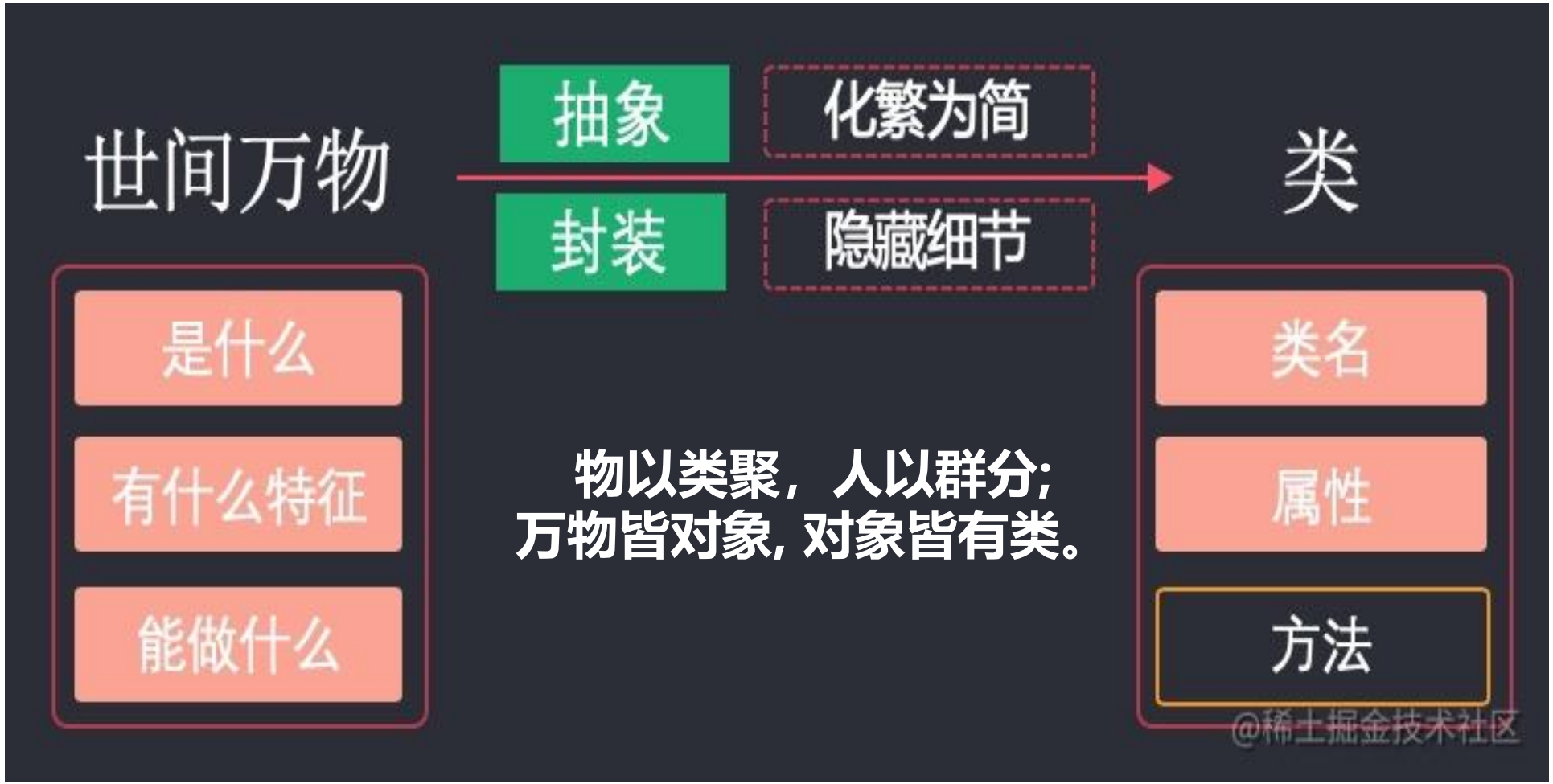
继承与派生
Inheritance & Derive

2025年3月24日

学而不厌 诲人不倦

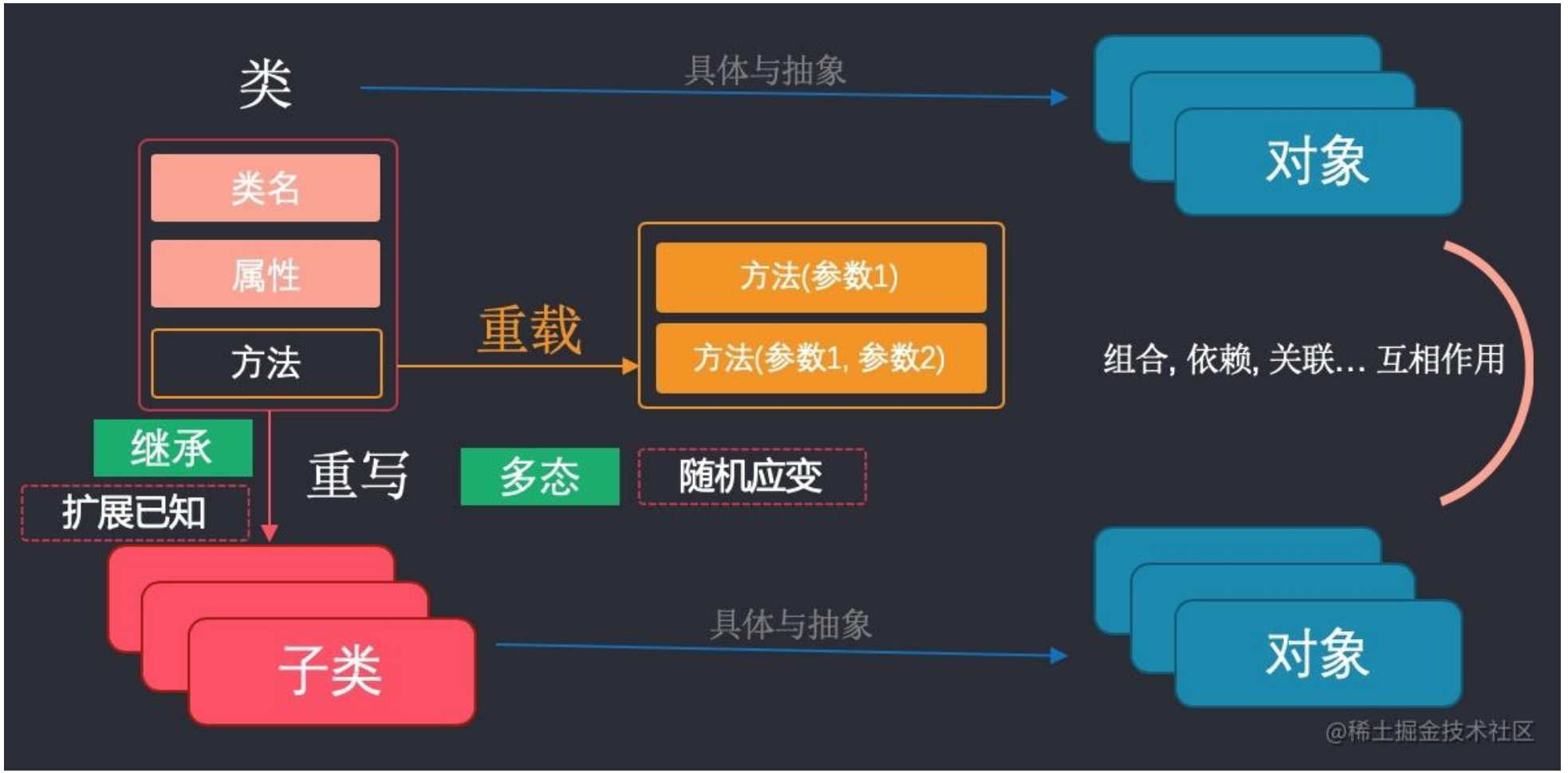
回顾

面向对象编程：抽象、封装、信息隐藏



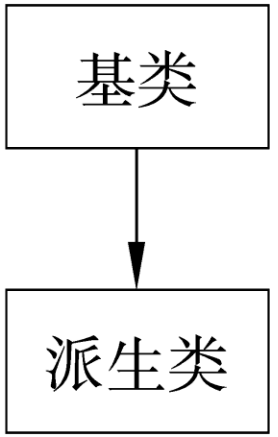
回顾

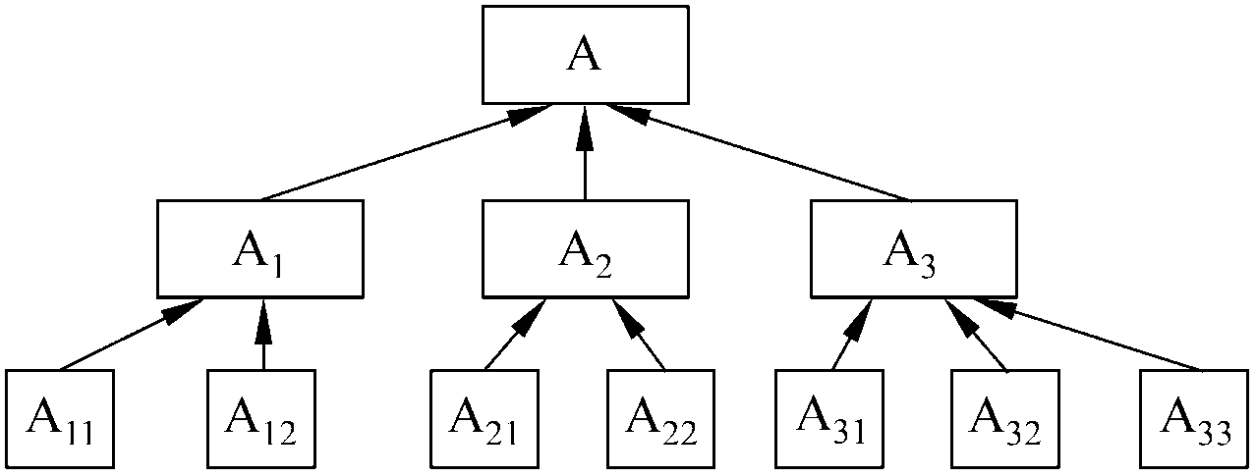
面向对象编程：模板、继承、多态



➢ 继承与派生的概念

继承是在已存在的类A 的基础上建立一个新类B。
类A称为基类或父类，类B 称作派生类或子类。
子类从父类获得其已有的特性，这种现象称作类的继承。
从另一个角度看从父类产生子类，称作类的派生。

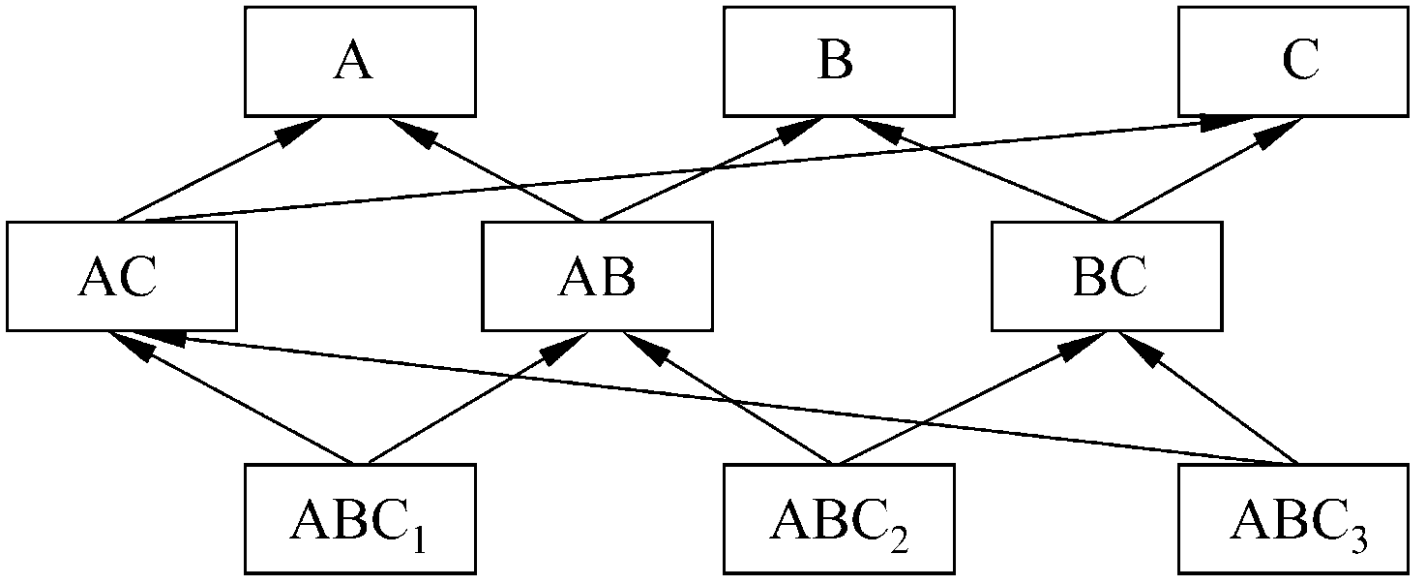




一个基类可以派生出多个派生类，每个派生类又可以作为基类再派生出新的派生类。
一个派生类只从一个基类派生，称作单继承。

➢ 继承与派生的概念

一个派生类也可从多个基类派生，也就是说一个派生类可以有两个或多个基类。一个派生类有两个或多个基类的称作多重继承。用图5.4表示。基类和派生类的关系可以表述为：**派生类是基类的扩充，而基类是派生类的抽象**。



➢ Demo01：继承

```
#include<iostream>
using namespace std;
//基类 Pelple
class People{
public:
  void setname(string  name);
  void setage(int age);
  string getname();
  int getage();
private:
  string  m_name;
  int m_age;
};
void People::setname(string name){ m_name = name; }
void People::setage(int age){ m_age = age; }
string People::getname(){ return m_name; }
int People::getage(){ return m_age;}
```

➢ Demo01：继承

```
//派生类 Student
class Student: public People{
public:
    void setscore(float score);
    float getscore();
private:
    float m_score;
};
void Student::setscore(float score){ m_score = score; }
float Student::getscore(){ return m_score; }
```

➢ Demo01：继承

```
int main(){
  Student stu;
  stu.setname("小明");
  stu.setage(16);
  stu.setscore(95.5f);
  cout<<stu.getname()<<“的年龄是”<<stu.getage()<<“，成绩是”<<stu.getscore()<<endl;
  return 0;
}
```

http://c.biancheng.net/view/2269.html

5.2 派生类的声明方式

➢ 1. 派生类的声明方式

使用派生类要先声明，声明的格式为

class 派生类名: [继承方式] 基类名

{ 派生类新增成员声明 };

继承方式包括：public、private、protected。如果省略，系统默认为private。

例：已经声明一个基类People，在它基础上通过单继承建立一个派生类student:

```
//基类 Pelple
class People{
public:
private:
};
```

```
//派生类 Student
class Student: public People{
public:
private:
};
```

2. 派生类的构成

派生类中的成员包括从基类继承过来的成员和自己增加的成员。继承基类成员体现了同一基类的派生类都具有的共性，而新增加的成员体现了派生类的个性。

(1) 从基类接收成员。 派生类将基类除构造函数和析构函数外的所有成员接收过来。

(2) 调整从基类接收的成员。 一方面可以通过继承方式改变基类成员在派生类中的访问属性，另一方面可以在派生类中声明一个与基类成员同名的成员屏蔽基类的同名成员

(3) 派生类增加成员实现扩充。

(4) 派生类需要自定义构造函数。

注意：屏蔽成员函数不仅要函数名相同，而且函数的参数也要相同，屏蔽的含义是用新成员取代旧成员。

➢ 2. C++继承时的名字遮蔽问题 Demo02

```
#include<iostream>
using namespace std;
//基类People
class People{
public:
    void show();
protected:
    string m_name;
    int m_age;
};
void People::show(){
    cout<<"嗨，大家好，我叫"<<m_name<<"，今年"<<m_age<<"岁"<<endl;
}
```

5.2 派生类的声明方式

➢ 2. C++继承时的名字遮蔽问题 Demo02

```
//派生类Student
class Student: public People{
public:
    Student(char *name, int age, float score);
public:
    void show(); //遮蔽基类的show()
private:
    float m_score;
};
Student::Student(string name, int age, float score){
    m_name = name;
    m_age = age;
    m_score = score;
}
void Student::show(){
     cout<<m_name<<"的年龄是"<<m_age<<", 成绩是"<<m_score<<endl;
}
```

➢ 2. C++继承时的名字遮蔽问题 Demo02

```
int main(){
    Student stu("小明", 16, 90.5);
    //使用的是派生类新增的成员函数，而不是从基类继承的
    stu.show();
    //使用的是从基类继承来的成员函数
    stu.People::show();
    return 0;
}
```

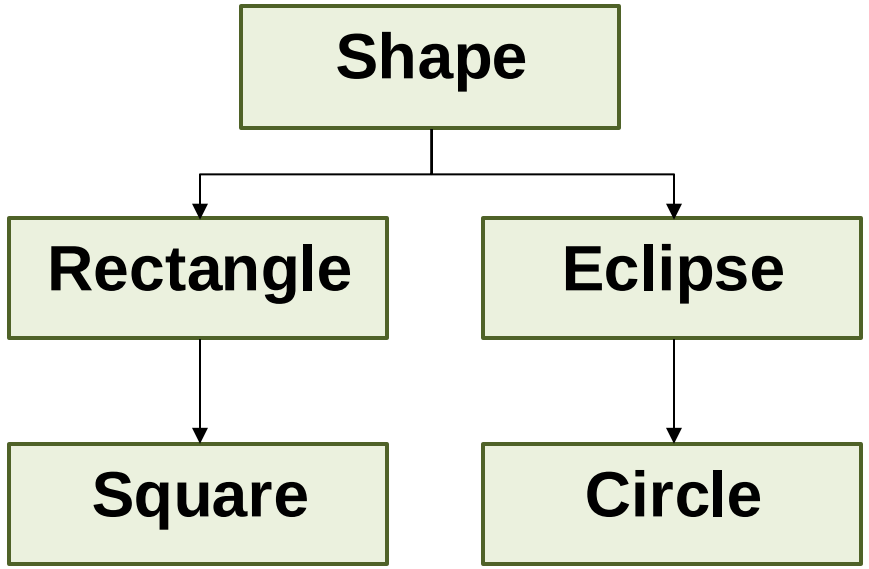
基类成员函数和派生类成员函数不构成重载

基类成员和派生类成员的名字一样时会造成遮蔽，这句话对于成员变量很好理解，对于成员函数要引起注意，不管函数的参数如何，只要名字一样就会造成遮蔽。

换句话说，基类成员函数和派生类成员函数不会构成重载，如果派生类有同名函数，那么就会遮蔽基类中的所有同名函数，不管它们的参数是否一样。

类的继承与派生

1. 设计基类shape
2. 设计基类的两个派生类Rectangle和Eclipse
3. 从Rectangle类派生Square类
4. 从Eclipse类派生Circle类



Thank You !

Q & A