第四章 | OOD

抽象类和接口

涉及到课本的章节:

- 第3章 类的封装、继承和多态
- 第4章 接口

第四章II 抽象类和接口

- 1. 抽象类
  - (1) 抽象类定义
- **2. 接口**
  - **(1) 接口定义**
  - **(2) 实现接口**
  - **(3) 接口继承**
- 3. 抽象类与接口

生活中的接口

秒懂易接

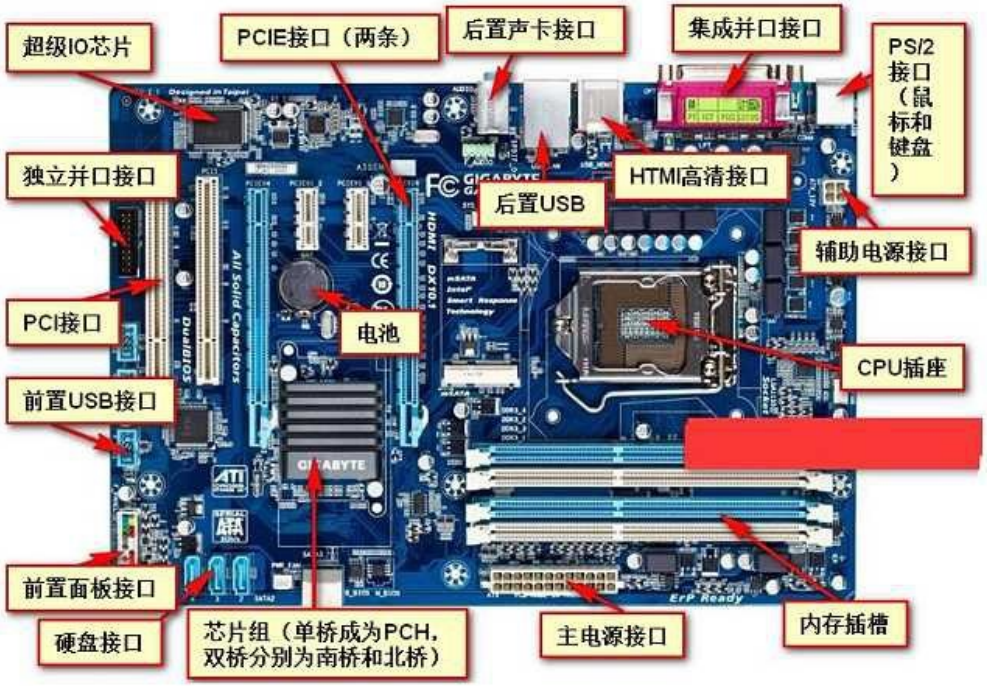

十字槽
加减槽（±）
一字槽
（开槽）
四方槽
（方插口）
一字方插口
内六角
内梅花
（内六角花形）
人字槽
二点槽


93PCS CR-V with Nickel Surface Treatment bits 4*28mm +9PCS Sockets

计算机的接口

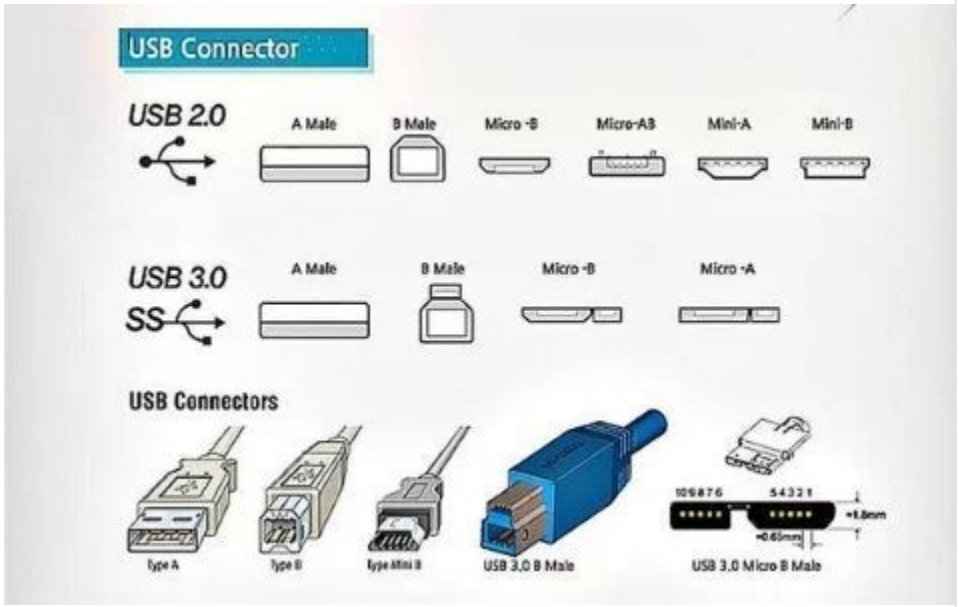
规格不一

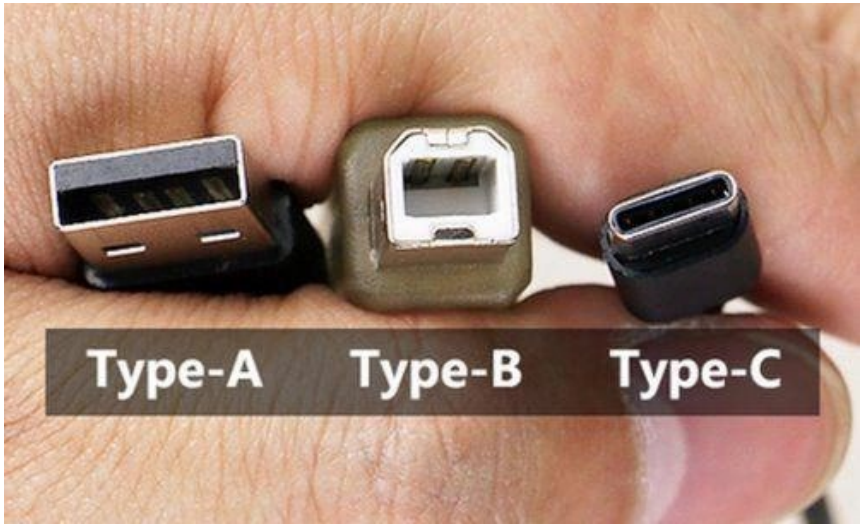

电源接口
2个键鼠接口
HDMI接口
VGA接口
COM接口
eSATA接口
2个USB2.0接口
2个USB2.0接口
网线接口
光纤接口
音频接口
VGA接口


超级IO芯片
PCIE接口（两条）
后置声卡接口
集成并口接口
PS/2接口（鼠标和键盘）
独立并口接口
后置USB
HTMI高清接口
辅助电源接口
PCI接口
电池
CPU插座
前置USB接口
前置面板接口
硬盘接口
芯片组（单桥成为PCH，双桥分别为南桥和北桥）
主电源接口
内存插槽

USB接口

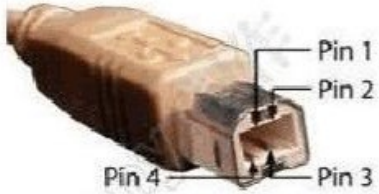
升级演变



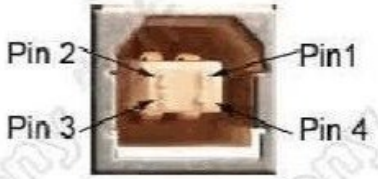


A型 USB接口（公口）

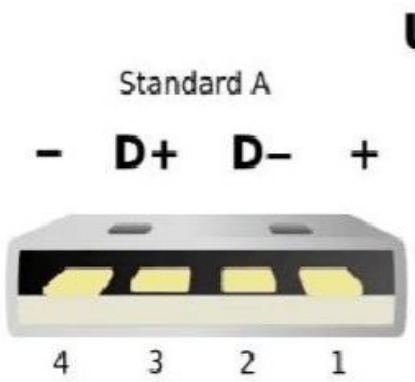
A型 USB接口（母口）

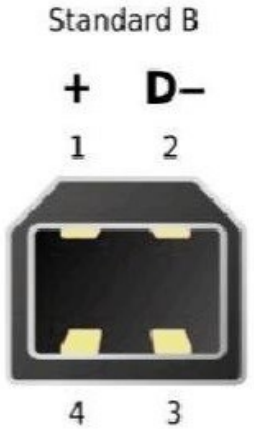


B型 USB接口（公口）



B型 USB接口（母口）



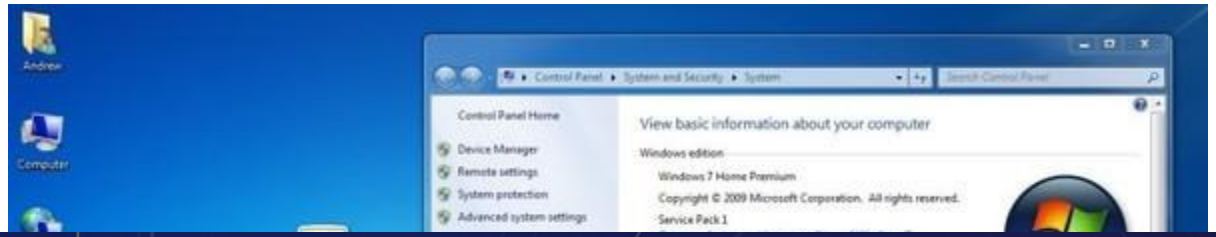


软件层面的常见接口

- API（Application Programming Interface，应用程序接口）是一些预先定义的函数，或指软件系统不同组成部分衔接的约定。

- 图形用户界面（Graphical User Interface，简称 GUI，又称图形用户接口）是指采用图形方式显示的计算机操作用户界面。

常见的接口——GUI


Control Panel Home
Device Manager
Remote settings
System protection
Advanced system settings
View basic information about your computer
Windows edition
Windows 7 Home Premium
Copyright © 2009 Microsoft Corporation. All rights reserved.
Service Pack 1

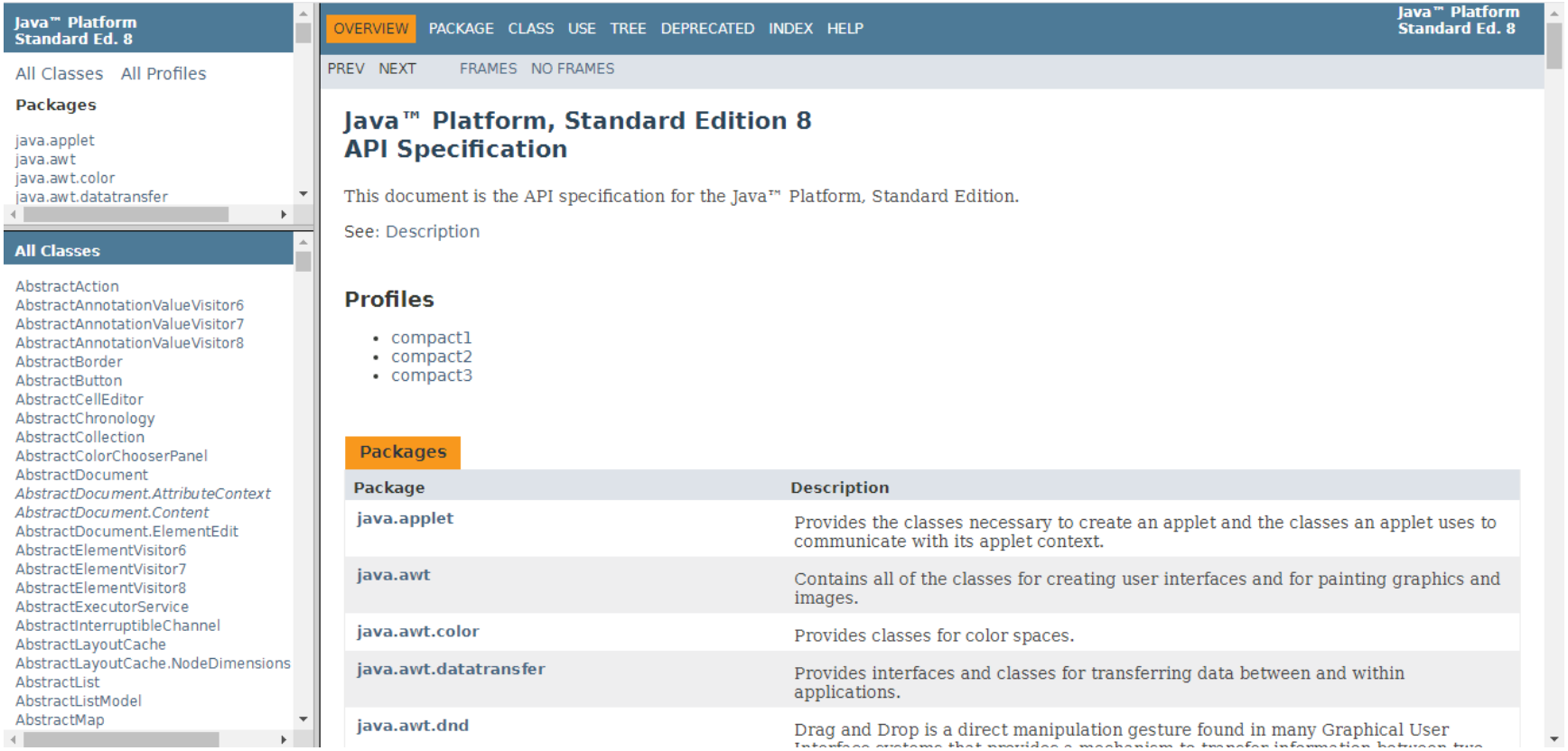

山东大学
Shandong University
信息化公共服务平台
English
账号登录
扫码登录
2019998000054
自动登录
忘记密码？
账号登录
温馨提示：
校外访问，请登录 Web VPN。
1.用户名为“职工号 / 学号”。若忘记密码或提示密码错误，请点此查看密码重置方法。
2.扫码登录前请先关注“山东大学微信企业号”。
3.浏览器请使用极速模式 (如何使用?)
4.建议浏览器：IE10+ 火狐 谷歌
山东大学 版权所有 Copyright@2017 信息化工作办公室
服务邮箱：passadmin@sdu.edu.cn

常见的接口——API

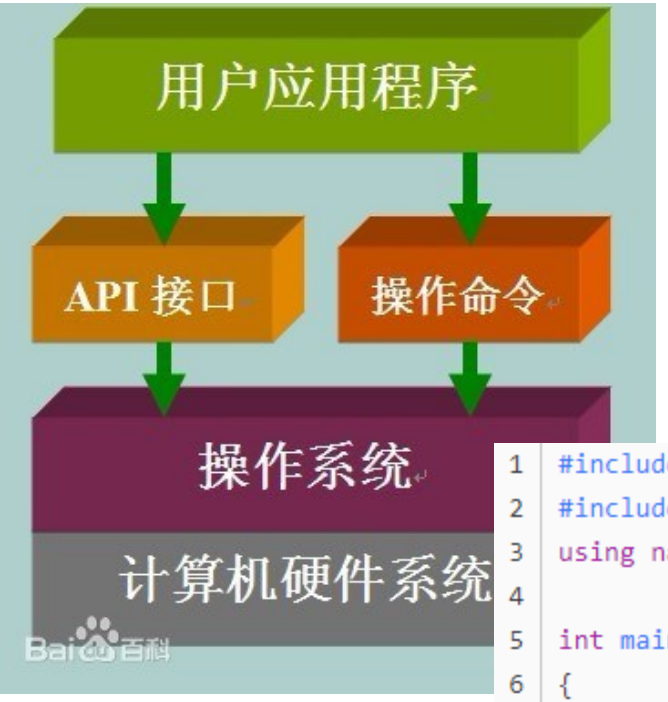
- 【Desktop】Windows API、Linux API
- 【Mobile】Android API、iOS API
- 【Application】地图API、即时通讯API
- 【Service API】百度API、API网站、微信小程序、支付宝小程序

- 【Java API】
- 编程语言语法规范

常见的接口——Java API



常见的接口——Windows API



- Win32接口

https://docs.microsoft.com/zh-cn/windows/win32/apiindex/windows-api-list

- VB接口

http://www.office-cn.net/t/api/web.htm

```
1  #include <iostream>
2  #include <Windows.h>//加上Windows.h头文件
3  using namespace std;
4
5  int main()
6  {
7      cout << "Hellow C++" << endl;
8
9      //弹出一个对话框函数，第一个参数为副窗口一般写NULL，第二个参数是消息的内容，第三个参数是标题
10     MessageBox(NULL, TEXT("Hellow C++"), TEXT("标题"), MB_OK);
11
12     system("pause");
13     return 0;
14 }
```

常见的接口——百度API

- http://ai.baidu.com/index/

常见的接口——功能变现

- 聚合数据https://www.juhe.cn/

接口的意义

- 将内部功能对外提供
- 对层内隐蔽实现细节-对层外提供使用约定

- 提供了方法声明与方法实现相分离的机制
- 实现运行时多态

(1)接口的定义

A 接口:

- **Java interface** 类中所有的成员方法都是抽象的;
- 接口由实现该接口的类来提供方法的具体实现。

B: Java接口组成

- 一个Java接口是由一组常量和抽象方法定义组成。

C 接口的定义:

```
[public] interface 接口名[extends 父接口]
{
    [public][static][final]类型 最终变量名=value;        //常量名
    ……;
    [public][abstract]返回类型 方法名(参数表);           //抽象方法说明;
    ……;
}
```

注: 常量定义中可以省略修饰符
(为了清晰或者可读性，可以加上修饰符)
抽象方法说明可以一般省略abstract,而加上public修饰符。
因为接口中没有抽象的概念

引申思考?

- Java接口的方法中可以有构造方法吗?

示例：定义接口

Sellable.java---------产品可出售标准的接口

```
public interface Sellable
 {
   String getUnits();
   double getPriceUnit();
   int getAmount();
   double getWeight();
  }
```

注：一个.java的文件最多只能有一个public的类或者接口；
当存在public的类或者接口时，文件名必须与这个类或者接口同名

第四章II 抽象类和接口

- 1. 抽象类
  - (1) 抽象类定义
- 2. 接口
  - (1) 接口定义
  - **(2) 实现接口**
  - (3) 接口继承
- 3. 抽象类与接口

2.实现接口

- 接口中的方法由类来实现.
- 语法:

  修饰符 **class** 类名**[extends** 父类名**]** **implements** 接口1

  [，接口2…]

  {……}
- 一个类可以实现多个接口

- 注：

  java只提供类的单继承，但是可以通过接口来实现多个接口。

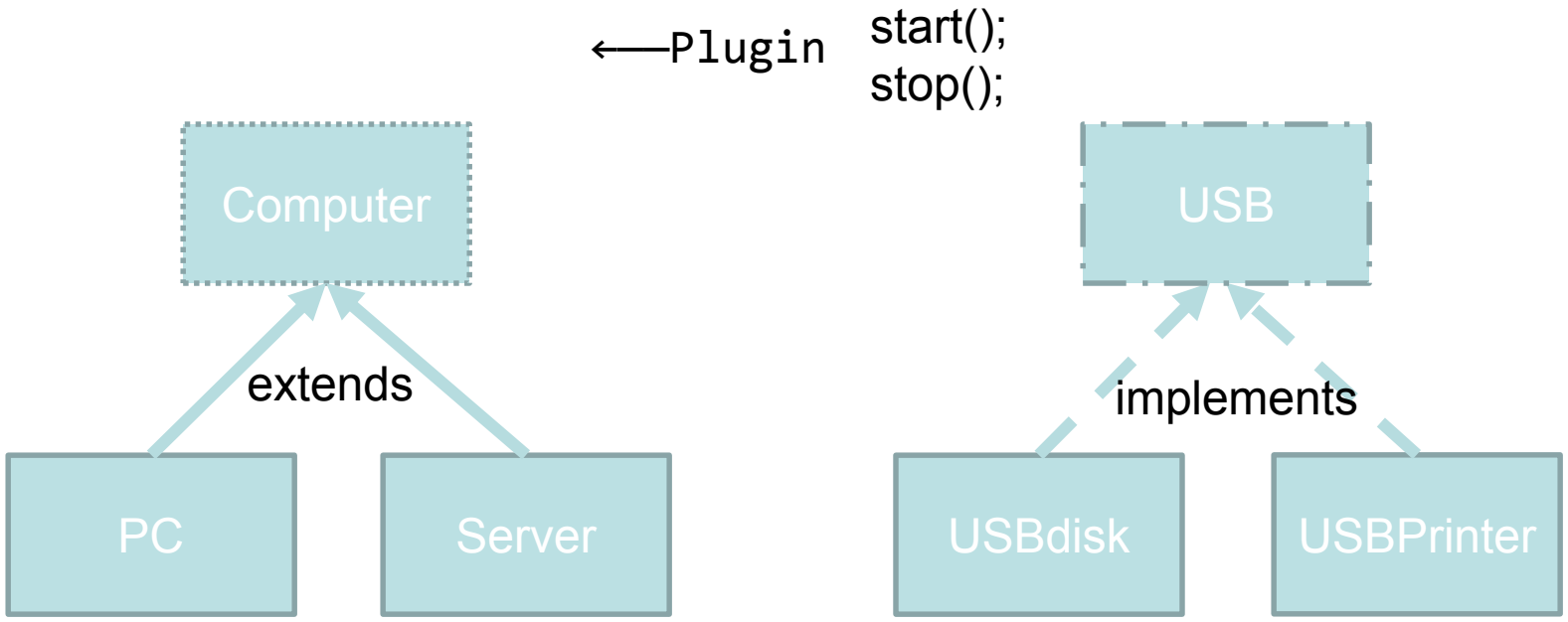
  如果一个类实现一个接口，则必须实现接口中的所有方法且方法必须声明为**public**的

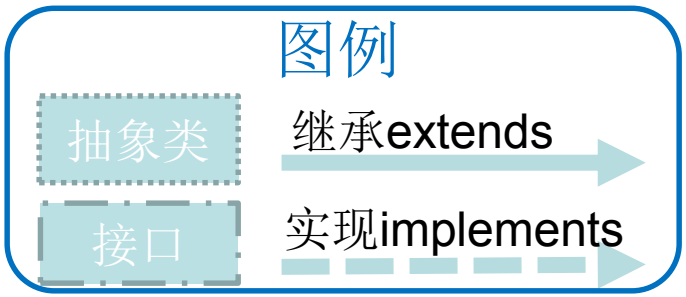
示例：接口的实现

JavaBook.java----编程语言书类

```
public class JavaBook implements Sellable{
        public String getUnits()
        {    return “Each”;}
        public double getPriceUnit()
        {   return 39.95;}
        public int getAmount()
        {    return 30;  }
        public double getWeight()
        {   return 4.5;}
}
```

抽象类、接口示例演示





cn.sdu.java.Interface.USBdevice

接口的多态

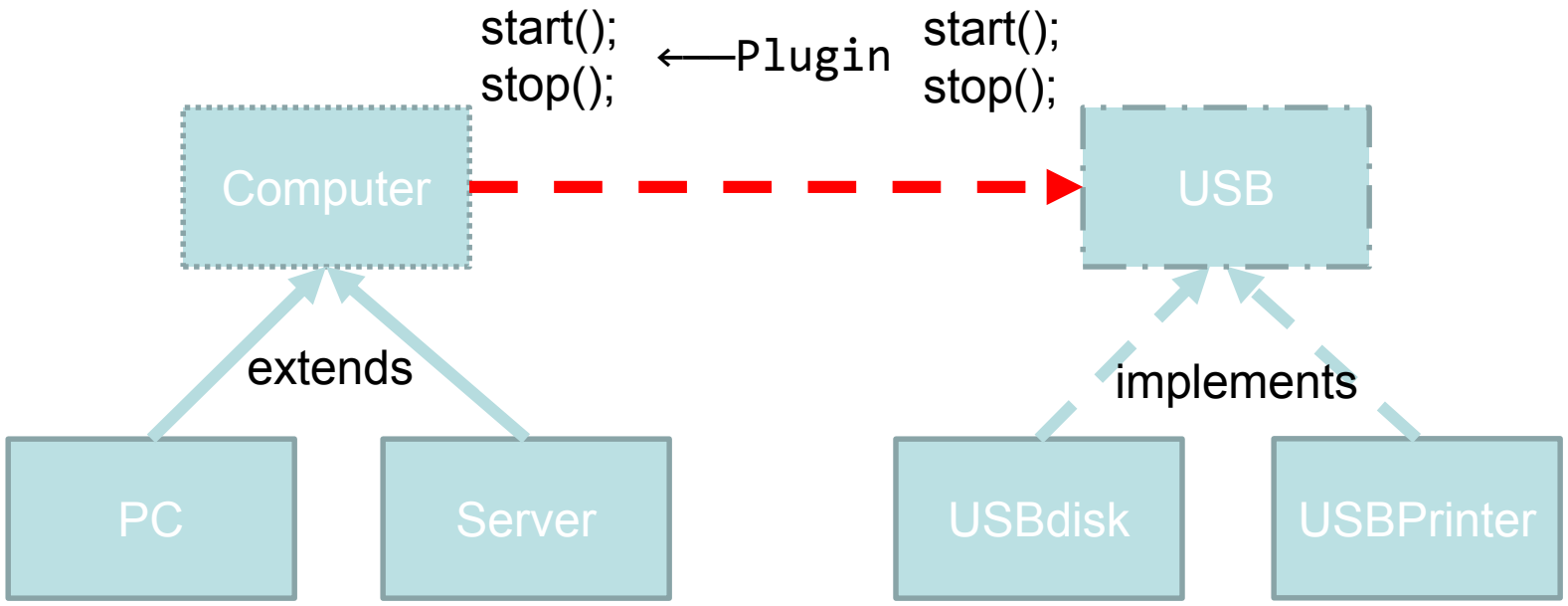

```
//电脑可以插入USB接口类设备（向上转型）
//只有符合USB接口的标准的类的对象（即只有实现这个接口的类的对象），才能被这个方法调用。
public void plugin(USB usbDevice){
    System.out.println("***** USB 设备插入 -----------
    usbDevice.start() ;
    if(usbDevice instanceof USBdisk)System.out.println
    if(usbDevice instanceof USBPrinter)System.out.pri
    usbDevice.stop() ;
    System.out.println("***** USB 设备拔出 -----------
    System.out.println();
}
```

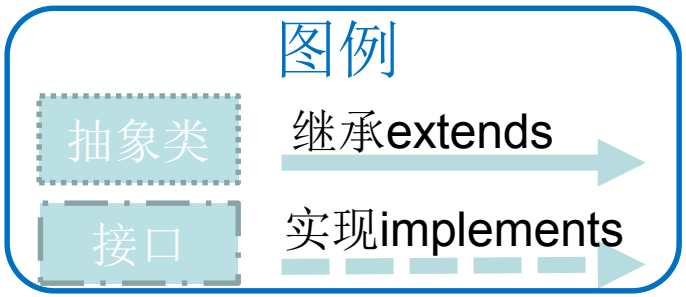
USB接口声明的start()和stop()抽象方法，在实现该接口的多个类中表现运行时多态

```
public static void main(String[] args) {
    Computer computer1 = new Computer_PC();
    USB usb1 = new USBdisk();
    computer1.plugin(usb1);
    computer1.plugin(new USBPrinter());
```

接口对象能够引用对其本身及其子接口进行实现的类及子类的实例。

抽象类、接口示例演示（交叉）





cn.sdu.java.Interface.USBdevice_Computer

插播: Java8 新特性

```
public  interface Test{
    //常量
    // 抽象方法
    // 非抽象方法
     default void mymethod(){……}
    //静态方法
    public static void staticMethod(){……}
}
```

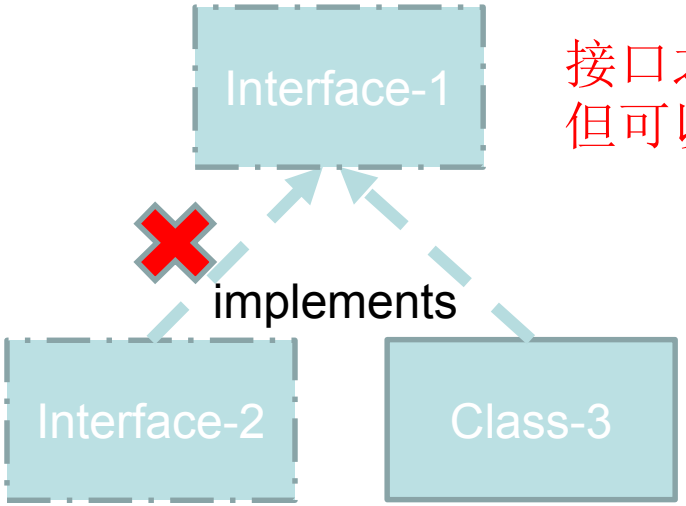
插播: Java8 新特性

```
public class A implements Test{
      //可以不重写实现mymethod
}
public class MainTest{
   public static void main(String[] s){
        A a=new A();
        a.mymethod();
}
```

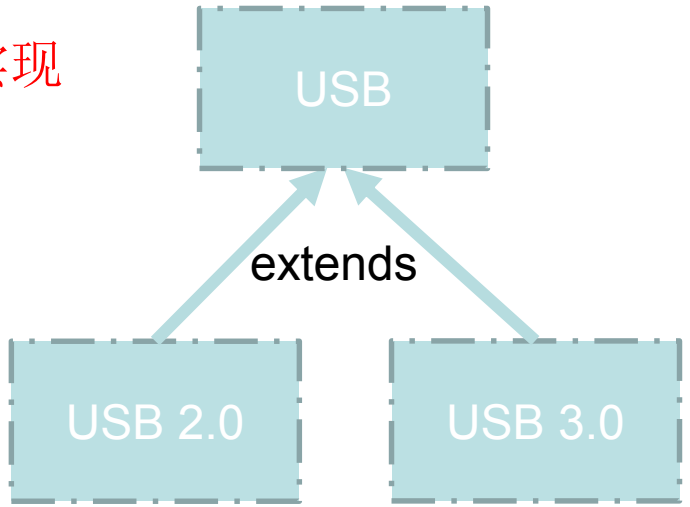
第四章II 抽象类和接口

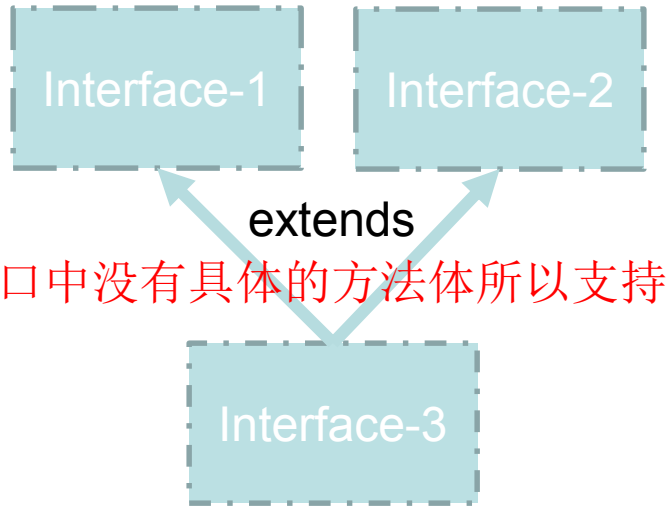
- 1. 抽象类
  - (1) 抽象类定义
- 2. 接口
  - (1) 接口定义
  - (2) 实现接口
  - **(3) 接口继承**
- 3. 抽象类与接口

接口之间的关系

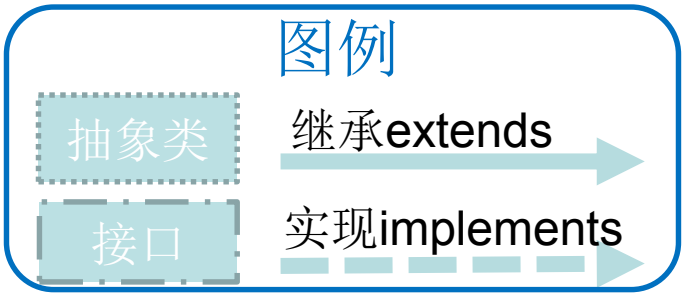


接口之间无法互相实现
但可以互相继承





接口中没有具体的方法体所以支持多继承



3.接口的继承示例

```
interface A
{
   int AA=1;
   void showa( );
}
```

```
interface C extends B
{
    int  CC=3;
    void showc( );
}
```

```
interface B extends A
{
  int BB=2;
  void showb( );
}
```

```
interface D extends C
{
   int DD=4;
   void showd( );
}
```

```
class E implements D
{    int e=5;
    public void showa( ){System.out.println("a="+a);}
    public void showb( ){System.out.println("b="+b);}
    public void showc( ){System.out.println("c="+c);}
    public void showd( ){System.out.println("d="+d);}
    public void showe( ){System.out.println("e="+e);}
}
```

```
class Test
{
    public static void main(String args[ ])
    {
        E   ee=new E( );
        ee.showa( );
        ee.showb( );
        ee.showc( );
        ee.showd( );
        ee.showe( );
    }
}
```

多层关系——接口继承后的实现

- 接口B继承接口A
- 接口C实现接口B也需实现接口A中的方法

- 接口肯定不能实现接口，但能继承接口。
- 接口虽然不能实现接口，但是抽象类可以实现接口
- 【思考】接口又是不是只能继承接口？可否继承抽象类？

接口支持多继承

- 接口是常量值和方法定义的集合。接口是一种特殊的抽象类。

- java类是单继承的。classB extends classA
- java接口可以多继承。Interface0 extends Interface1, Interface2, interface3……

接口支持多继承

- 不允许类多继承的主要原因是，如果A同时继承B和C，而B和C同时有一个test方法，A如何决定该继承那一个呢？被多继承的两类如有相同方法有可能不一样产生冲突
- 但接口不存在这样的问题，接口全都是抽象方法继承谁都无所谓，所以接口可以继承多个接口。被多实现的两接口即使有相同方法也肯定一模一样只有空壳不会冲突

接口的继承与实现

- 接口继承接口可以不实现父接口中的方法，可以声明自己的新方法
- 抽象方法只能定义在抽象类中，抽象类实现接口，可以不实现接口中的抽象方法
- 类实现接口时，一定要实现接口中声明的方法及其父接口中的方法

第四章II 抽象类和接口

- 1. 抽象类
  - (1) 抽象类定义
- 2. 接口
  - (1) 接口定义
  - (2) 实现接口
  - (3) 接口继承
- 3. 抽象类与接口

(1)二者区别

- 相同点
  - 都包含抽象方法
  - 都不能被实例化
  - 都是引用数据类型

不同点

- 约定:
  抽象类 约定了多个子类共同使用的方法.
  接口 约定了多个互不相关的类之间共同使用的方法
- 继承性:
  抽象类与子类之间采用单父节点机制（单继承）
  一个类实现多个接口则可以实现多父节点（多实现）

结构:

抽象类中可以包含非抽象方法,也可以声明构造方法.
接口中的方法全是抽象方法,不能声明构造方法

- 访问权限:
  抽象类具有和普通类一样的访问权限
  接口和类也是一样的访问权限,但是成员只能是public权限
- 成员变量:
  抽象类中可以声明成员变量,子类可以对该成员变量赋值
  接口只能是常量

抽象类与接口的关系

	继承类	实现接口	被继承	被实现
普通类	✔	✔	✔	✖
抽象类	✔	✔	✔	✖
接口	✖	✖	✔	✔

- 只有接口可以被实现
- 接口不能继承类
- 接口之间无法互相实现，但可以互相继承

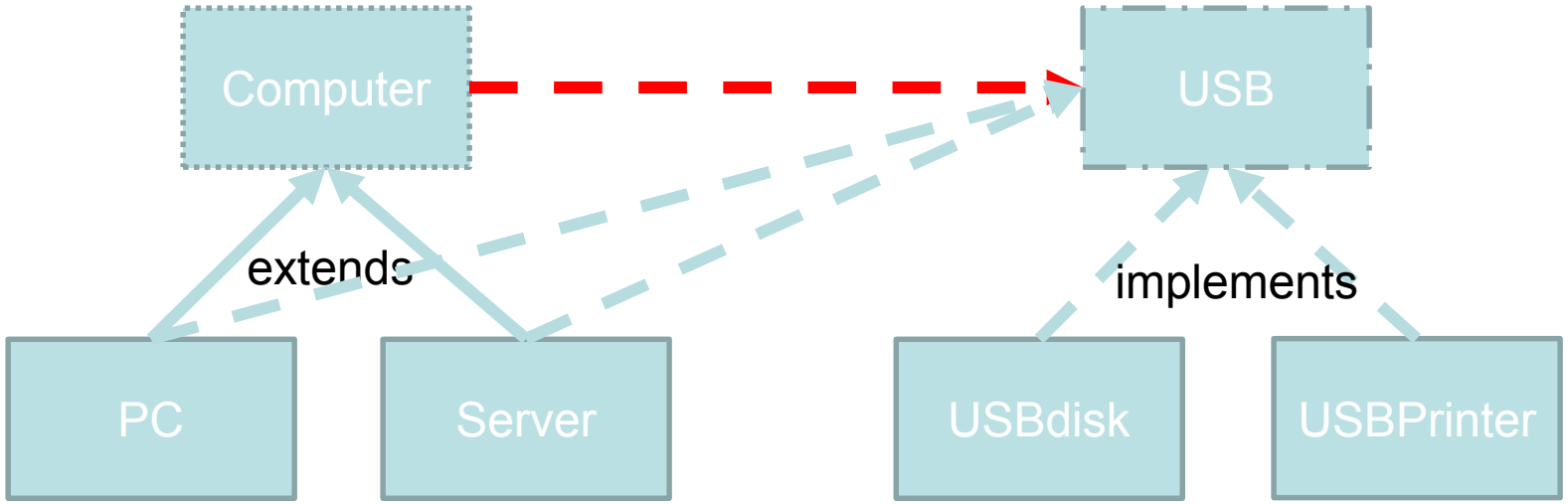
抽象类与接口的关系（继承）

主语 \ 宾语	普通类	抽象类	接口
普通类	✔	✔	✖
抽象类	✔	✔	✖
接口	✖	✖	✔

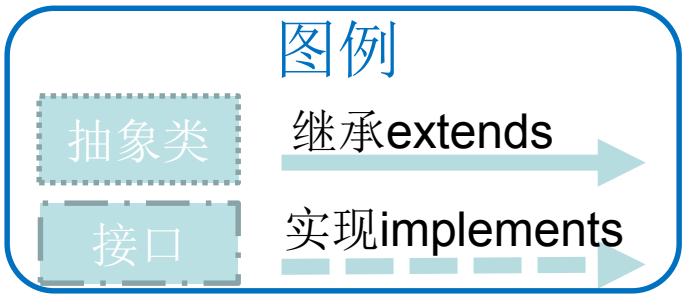
抽象类与接口的关系（实现）

主语 \ 宾语	普通类	抽象类	接口
普通类	✖	✖	✔
抽象类	✖	✖	✔
接口	✖	✖	✖

继承类的同时实现接口



- 继承父类的共性特征
- 实现接口的共同标准



【思考】一个类能否继承一个接口？

- 继承关系的父与子必须是同类

- 父与子都是类
- 父与子都是接口

总结

- **抽象类**——禁止实例化的类
  - 构造方法、抽象方法、**非抽象方法、成员变量**
  - 有抽象方法必须定为抽象类，抽象类中不一定要有抽象方法，子类必须对所有抽象方法进行实现
  - 可被单继承

- **接口**——特殊的抽象类但严格说已脱离类
  - 构造方法、抽象方法、非抽象方法、常量
  - 可被多实现、可多继承

【思考】接口是否继承Object？

- 接口是一种特殊的抽象类。
- 接口中都是抽象方法，如果继承Object是否继承了Object中的通用方法，如toString()、equals()等

- 如果答案为是，那么类实现多个接口的时候变相实现了类的多继承？

父类索引javap -verbose