## 类

### 类的加载顺序

静态变量、静态代码块（从上到下的顺序加载）

类的非静态变量，非静态块（从上到下的顺序加载）

构造函数

### 抽象类与抽象方法

1. 抽象方法必须在抽象类中定义，也就是说有抽象方法的类一定是抽象类。
2. 抽象类作为一个父类，可以定义抽象方法也可以定义一般方法
3. 子类继承一个抽象父类，一定、必须要实父类的抽象方法

### 抽象类与接口的区别

1、抽象类使用abstract修饰；

2、抽象类不能实例化，即不能使用new关键字来实例化对象；

3、含有抽象方法（使用abstract关键字修饰的方法）的类是抽象类，必须使用abstract关键字修饰；

4、抽象类可以含有抽象方法，也可以不包含抽象方法，抽象类中可以有具体的方法；

5、如果一个子类实现了父类（抽象类）的所有抽象方法，那么该子类可以不必是抽象类，否则就是抽象类；

6、抽象类中的抽象方法只有方法体，没有具体实现；

接口

1、接口使用interface修饰；

2、接口不能被实例化；

3、一个类只能继承一个类，但是可以实现多个接口；

4、接口中方法均为抽象方法；

5、接口中不能包含实例域或静态方法（静态方法必须实现，接口中方法是抽象方法，不能实现）

### 类的静态属性

#### 静态变量定义

static 变量名；

#### 静态变量的访问

在类内部：self::name;

在类外部：类名::name; 属性是public才可访问

#### 静态方法的定义

访问修饰符 static function 函数名(){}

Public static function test(){}

#### 静态方法的访问

在类外部：类名::静态方法名，对象名::静态方法名，对象名::静态方法

1.Person::test(); //优先使用这个

2.$p=new Person();

$p->test();

3.$p1::test(); //不推荐

在类内部：self:: 静态方法名 ,类名:: 静态方法名,$this->静态方法名

Self::test();//优先使用

Person::test();

$this->test();

#### Self与this区别

1. 使用方式不同

Self::

$this->

1. self是类范畴指向类，$this是对象实例指向对象实例

#### new self( )

new self() 一般在类内部使用，作用是对自身类实例化

## 继承

### 什么是继承

子类继承父类后拥有父类的所有属性和方法，但是父类中的private属性和方法只能拥有无法访问及子类是无法直接输出父类的private属性

### 子类继承父类的执行顺序

当实例化一个子类对象时，执行顺序为

父类静态变量、父类静态代码块（从上到下的顺序加载）

子类静态变量、子类静态代码块（从上到下的顺序加载）

父类的非静态变量、父类的非静态代码块（从上到下的顺序加载）

父类的构造方法

子类的非静态变量、子类的非静态代码块（从上到下的顺序加载）

子类的构造方法

未完待续