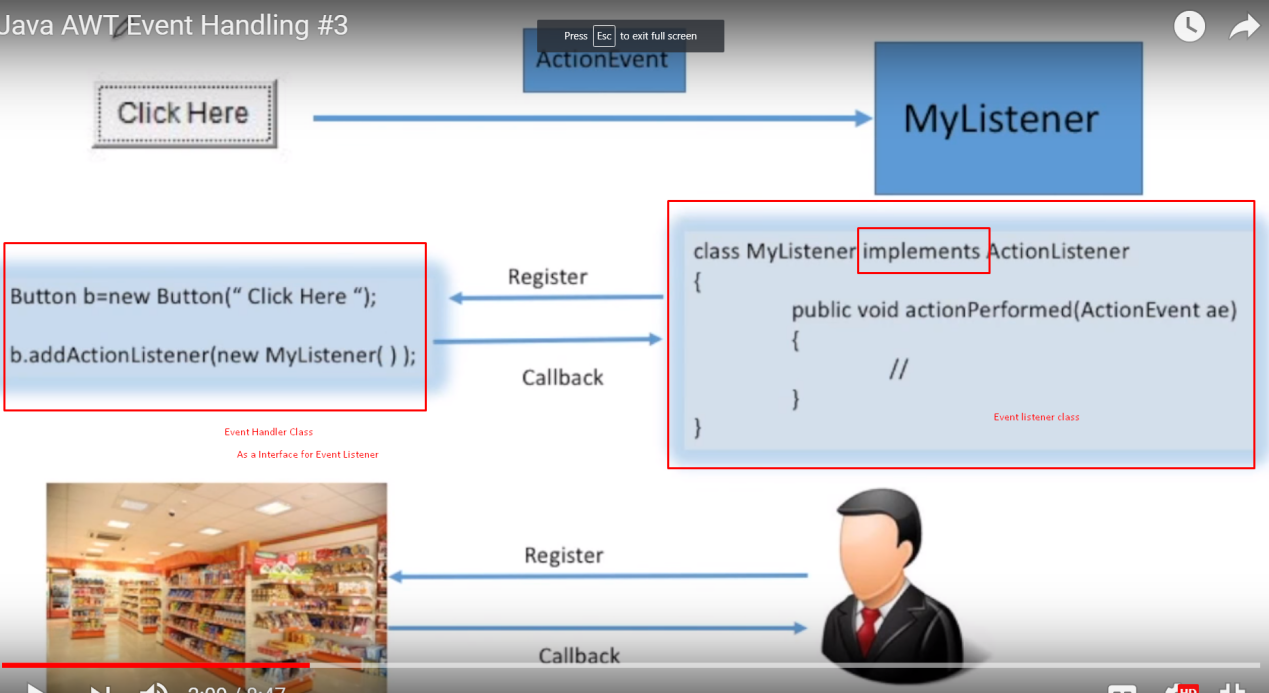
**Event handler and Event listener**



**AWT : Abstract Window Toolkit**

一套java提供写图形界面的API

**Design Pattern:别人之前写过的设计模式(代码套路样板):**

Java一共有20多种设计模式，归为三大类->

* Creational
* Structural
* Behavioral

**Method signature = return type + method name + parameter**



**I/O -> Input API and method:**

处理input常用的API/stream classes:

BufferReader

Scanner

处理input常用method:

System.in

hasNextxxx()

inputStringReader()

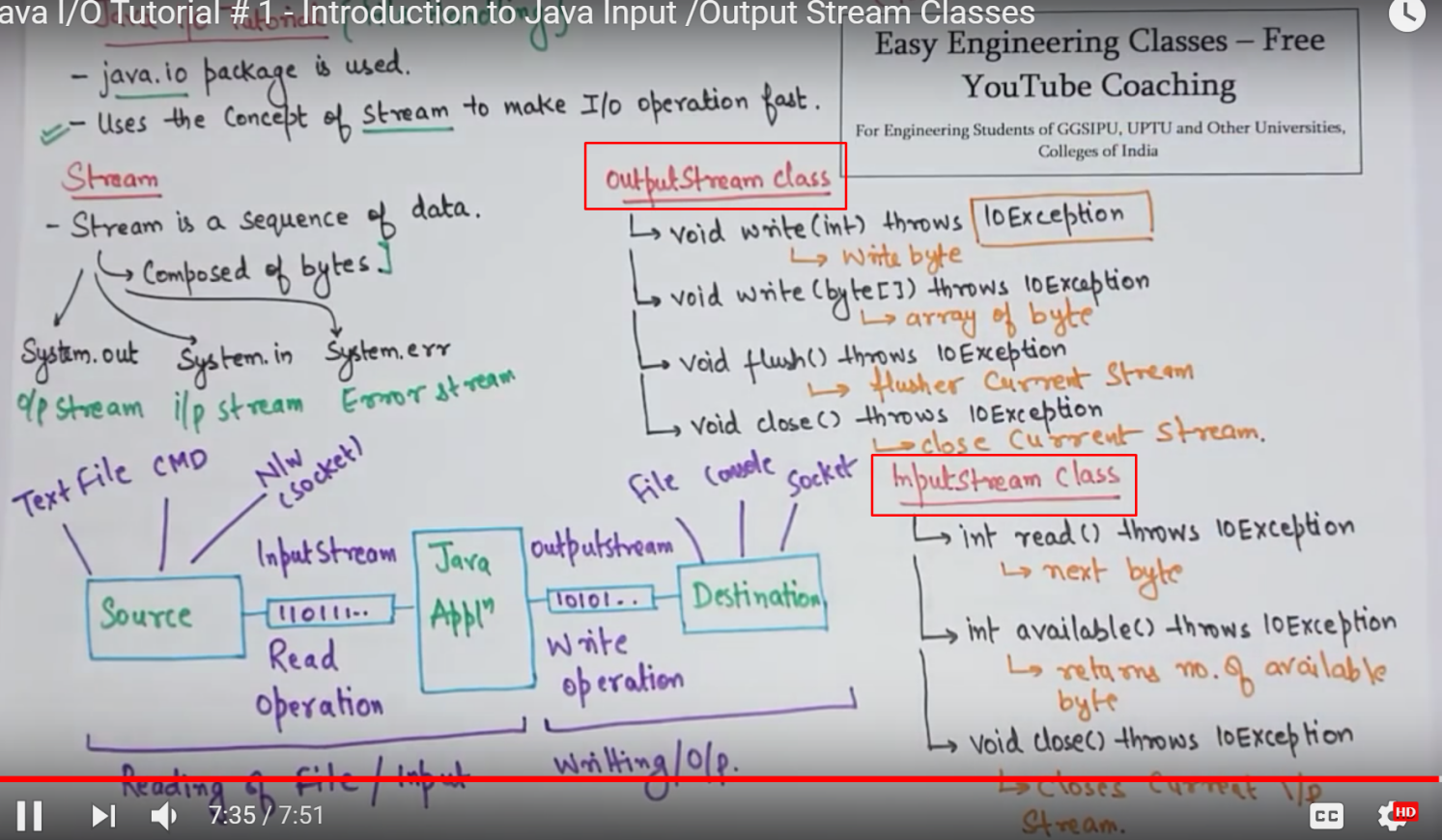
BufferReader()

Integer.ParseInt()

tokenizer 分词器/编译器

delimiter 字段分隔符号

underlying character stream 底层字符流



————————————————————

**Exception Handling** //Exception = Error

=>

Try{ }

catch (Exception e)//Exception means catch any errors occurs, e is the found error type

{ }

.exist(1) //退出并返回1给系统，1代表程序

异常，类推0代表正常运行

throws : suppress the error but not handling the error

————————————————————

**Interface = API = 一个可以被调用的接口包(里面可以有规定的方法和常量):**

Inside Interface is bunch of public abstract method and public static final data(constant).

调用interface的关键字: implements

**abstract classes and interface:**

abstract method : a method only with signature but not fully implemented

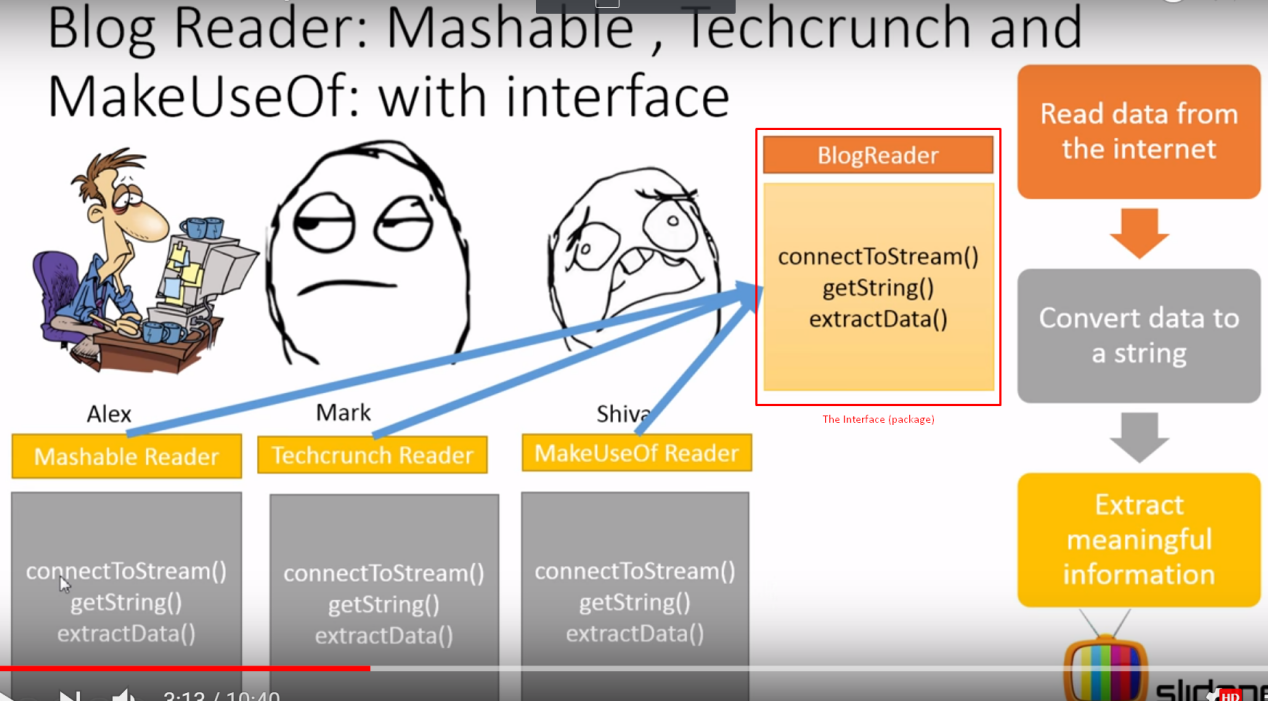
abstract class : a class at least contain one abstract method(can have concrete method inside as well)

interfaces : a collection of abstract methods which can be implemented by other classes(contact with the subclass to implement/override by the abstract methods)

**Difference between abstract classes and interface:**

1. abstract class can contain both abstract and concrete method ; while interface only can contain abstract method
2. abstract class can make empty method which AVALIABLE for subclass to override; while every method interface MUST be override by the subclass(interface is 100% abstraction)
3. performance speed abstract class is faster than using interface

联想问题: what affect program run-time efficiency?



**abstract classes:**



The reason of using abstract classes think about :

implement a calculation system prototype for 4 different shops , and the incomplete method only shall be complete by the subclass which it’s the individual shops base on their specification.

p58

**多态**

static biding

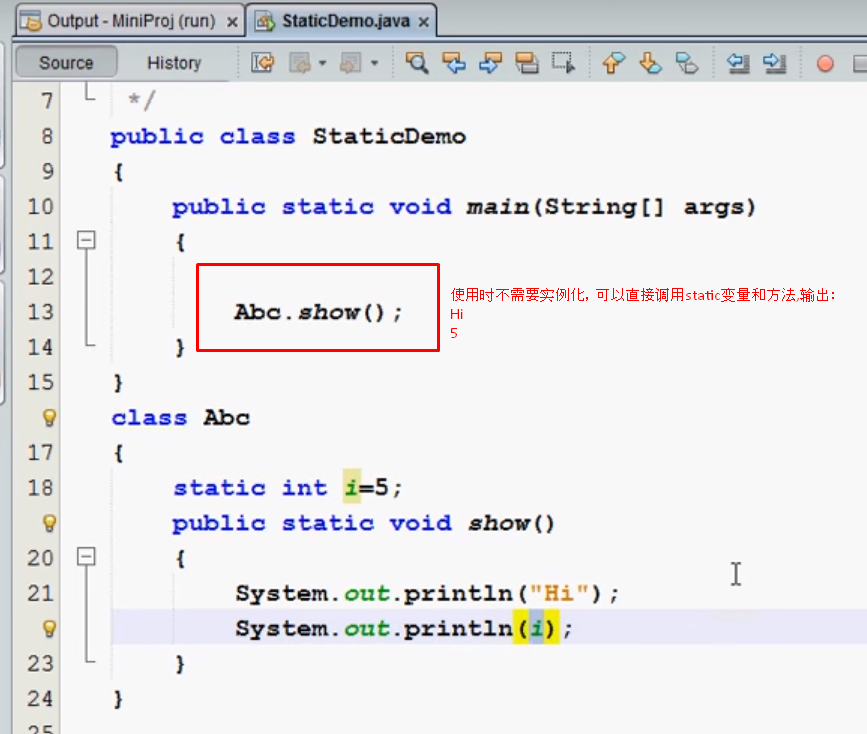
dynamic biding

overriding

run time and compile time

**static静态 :**

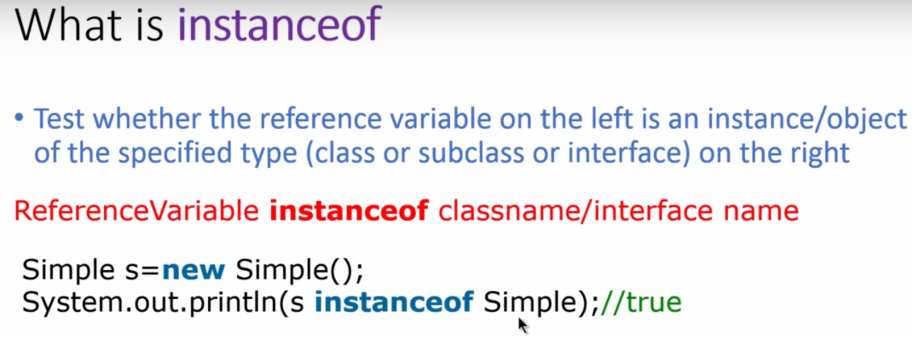
不依附于任何类独立存在的方法或变量，可以直接调用而不需要实例化



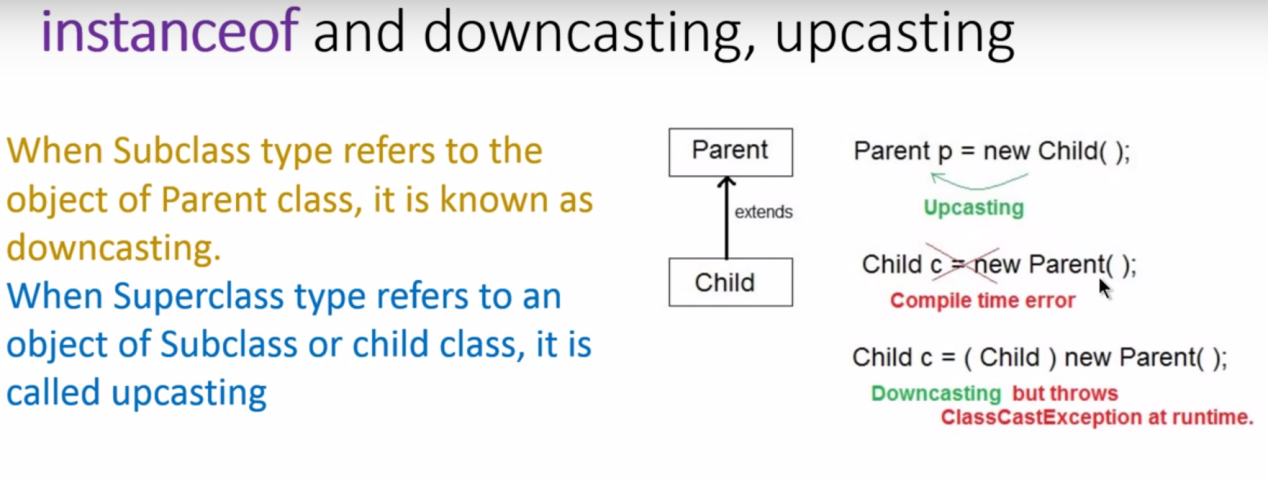
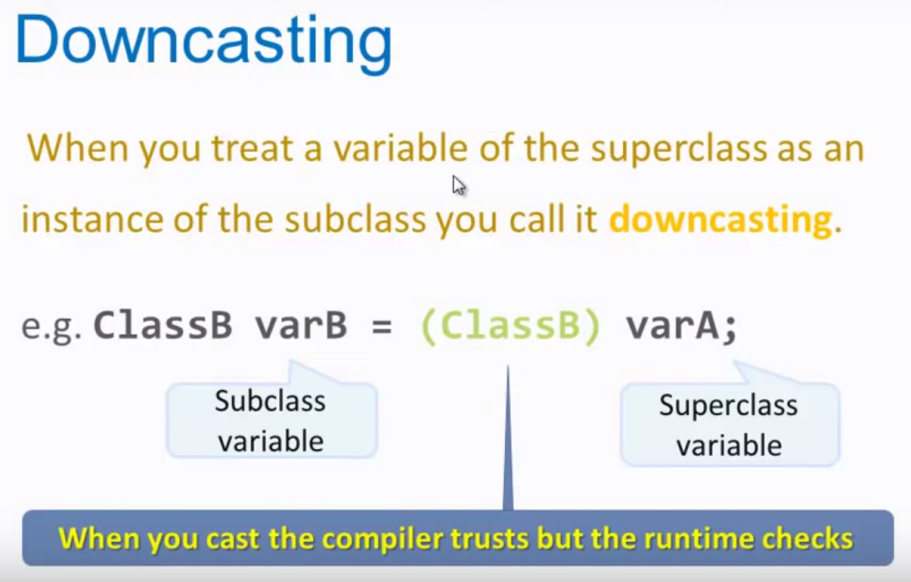
**compile time vs run time:**

编译时(执行之前) vs 执行时 ...

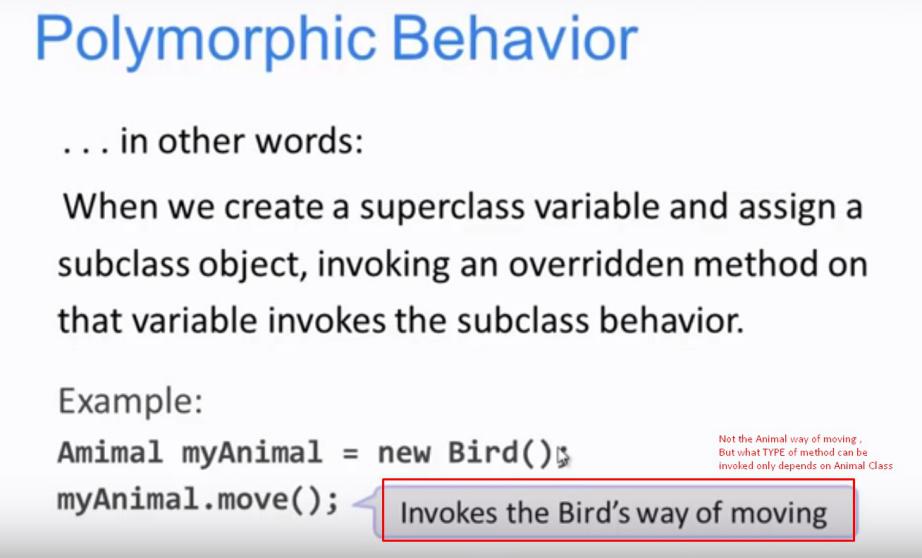
**instanceof判断:**



**upcasting and downcasting：  
Upcasting and Downcasting(向上向下强制转型是java中多态概念的重要体现)**

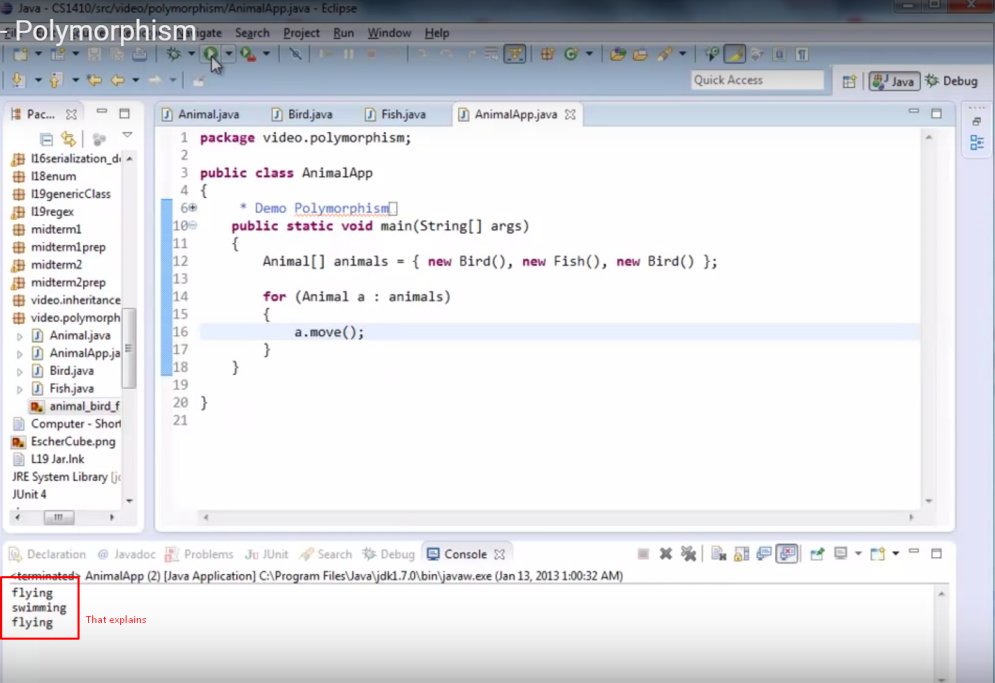


**多态中method调用:**

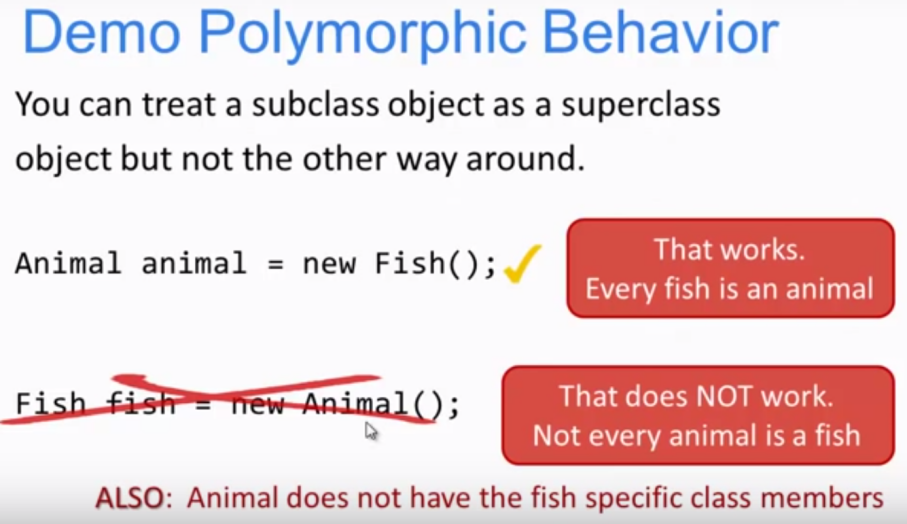


Because that’s how **dynamic biding** (checked by compiler) works in polymorphism

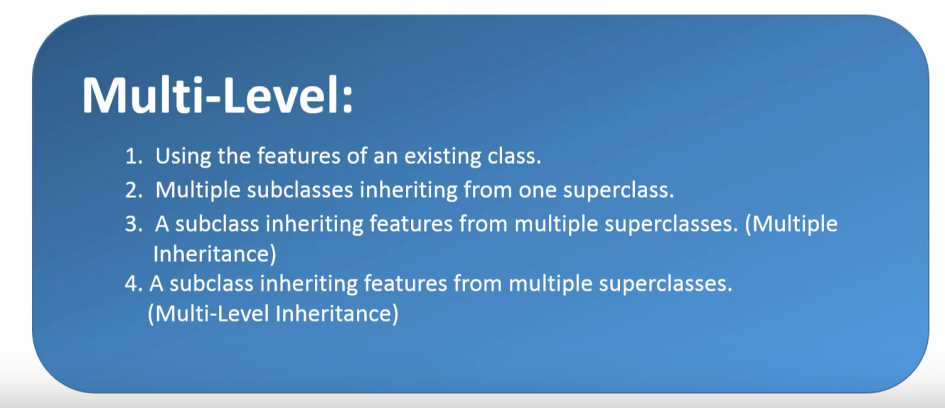
subclass method can rewrite superclass method , that’s the reflection of polymorphism.

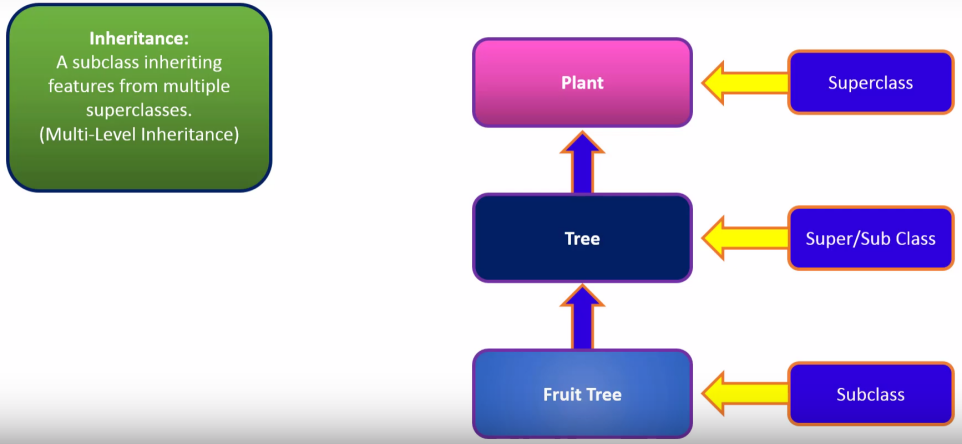


**多态中实例化对象的(等级)限制:**



**多级继承：**

=>



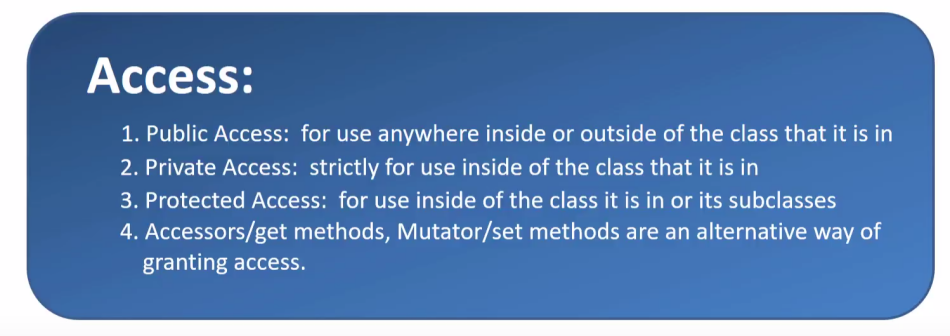
Categories of method in java:

Set method -> .addtime() //设定

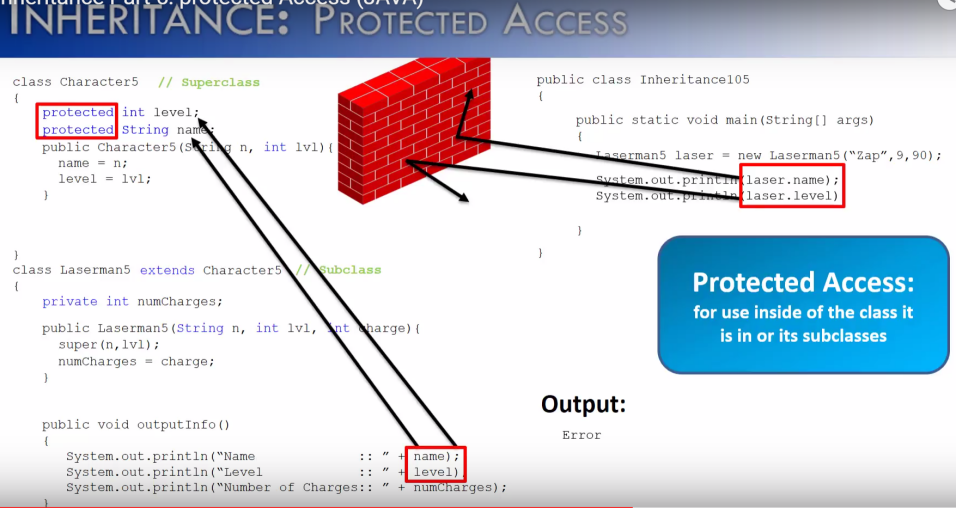
Access method -> .gethour() //获取

Mutator method -> .setTime() //改变

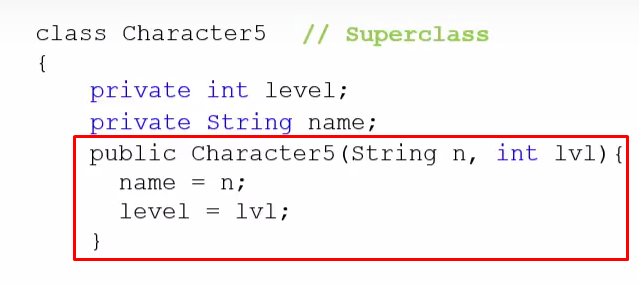
Constructor method //用来初始化类中的变量的函数



**Protected variable: grant access to only the subclasses.**



**构造函数：**



框中的为构造函数(constructor)，用来初始化类中的变量的函数，构造函数名称和类名称相同

**Java Inheritance ：关键字extends**



例子：

