#### <http://www.cnblogs.com/lanxuezaipiao/p/4153070.html> 别人的学习笔记

<http://blog.csdn.net/chjttony/article/details/6648534> good

#### 1.7**伴随多态的可互换对象。**

##### **­——（即父类的引用指向子类的对象，多态）、**

|  |
| --- |
| 在面向对象编程中，子类中拥有和父类相同方法签名的方法称为子类方法覆盖父类方法，当调用子类方法的某个操作时，不必明确知道子类的具体类型，只需要将子类类型看作是父类的引用调用其操作方法，在运行时，JVM会根据引用对象的具体子类类型而调用应该的方法，这就是多态。  多态的基础是java面向对象编程的晚绑定机制。编程中有如下两种绑定机制：  (1).早绑定：一般在非面向对象编程语言中使用，在程序编译时即计算出具体调用方法体的内存地址。  (2).晚绑定：面向对象编程语言中经常使用，在程序编译时无法计算出具体调用方法体的内存地址，只进行方法参数类型和返回值类型的校验，在运行时才能确定具体要调用方法体的内存地址。 |

Eg.List list = new ArrayList();

List.add(…);

多态的基础：动态绑定。

1. 在编译期，编译器是无法准确找到哪段代码被执行，通过一段特殊的代码代替绝对地址的调用，在运行时期，才能够确定代码的位置。
2. 后期绑定(面向过程的叫法，java中称为动态绑定)，运行时才知道哪段代码被执行。

前期绑定（面向过程的叫法，java中称为静态绑定），编译期就已经加载到内存。

方法绑定：一个方法被调用时该方法关联其方法体的过程。

1. 动态绑定，将函数调用和真正被执行的代码关联起来。编译期间，真正被调用的对象不知道被调用，运行期间，才明确那个对象的函数被调用。

注意：

|  |
| --- |
| * 动态绑定的范畴只是方法，不包含成员变量，子类同名的变量是不会覆盖父类的变量的；eg. Peson p = new Ling(); p.name = ? 注意这时输出的是person的name值，不是Ling的name值。 * java中的变量都是静态绑定的； * java中的构造方法 以及private,static,final类方法的调用都是静态绑定的。 |

**补充：**

<http://blog.csdn.net/sureyonder/article/details/5569617> [JAVA动态绑定的内部实现机制](http://blog.csdn.net/sureyonder/article/details/5569617)

<http://blog.csdn.net/zdwzzu2006/article/details/5979923> 静态绑定和动态绑定（good）

 向上转型（upcasting）

1. 多态的表现形式
   1. 方法参数：父类接收子类的对象。
   2. 创建对象：父类的引用指向子类的对象。

#### **1.8单根继承机制**

##### —即类的单继承，所有类最终都继承自单一的基类(object）

|  |
| --- |
| 4.java单继承的优点：  相比于C++的多继承，java只支持类的单继承，java中的所有类的共同基类是Object类，Object类java类树的唯一根节点，这种单继承有以下好处： |

单根继承机制的优点：

@保证所有对象都有某些共性功能。Eg.getClass(),toString()，对于JVM虚拟机对所有的类进行系统级的操作将提供方便，所有的java对象可以方便地在内存堆栈中创建，传递参数也变的更加方便简单。

@ java的单继承使得实现垃圾回收器功能更加容易，因为可以确保JVM知道所有对象的类型信息。反射and getclass（）；

1.9