## Get和Post的区别

1.get是从服务器上获取数据，post是向服务器传送数据，

2.get传送的数据量较小，不能大于2KB。post传送的数据量较大，一般被默认为不受限制。

3.get安全性非常低，post安全性较高。但是执行效率却比Post方法好。

4.在进行文件上传时只能使用post而不能是get。

## JavaSE JavaEE JavaME区别

是什么：

Java SE=Java Standard Edition=j2se = java 标准版

Java EE=Java Enterprise Edition=j2ee= java 企业版

Java ME=Java Mobile Edition=j2me = java移动版

特点：

SE主要用于桌面程序（swing）,控制台开发(main程序)。

EE企业级开发(JSP,EJB,Spring MVC,Struts,hibernate,ibatis等)，

用于企业级软件开发，网络开发，web开发。

ME嵌入式开发(手机,小家电，PDA)。[苹果的ios，黑莓]

三者之间的关系：

Java SE（Java Platform, Standard Edition，Java标准版）就是基于JDK和JRE的。

Java SE为Java EE提供了基础。

Java EE除了基于我们这个所谓的Java SE外，还新加了企业应用所需的类库

## JDK常用的包

　java.lang： 这个是系统的基础类，比如String、Math、Integer、System和Thread， 提供常用功能。

　java.io： 这里面是所有输入输出有关的类，比如文件操作等

　java.net：这里面是与网络有关的类，比如URL,URLConnection等。

　java.util：这个是系统辅助类，特别是集合类Collection,List,Map等。

　java.sql： 这个是数据库操作的类，Connection, Statememt，ResultSet等

## JDK JRE JVM的区别：

Jdk【Java Development ToolKit】就是java开发工具箱， JDK是整个JAVA的核心里边包含了jre，它除了包含jre之外还包含了一些javac的工具类，把java源文件编译成class文件，java命令是用来运行这个程序的，除此之外，里边还包含了java源生的API，java.lang.integer在rt的jar包里边【可以在项目中看到】，通过rt这个jar包来调用我们的这些io流写入写出等

JDK有以下三种版本：

J2SE，standard edition，标准版，是我们通常用的一个版本

J2EE，enterpsise edtion，企业版，使用这种JDK开发J2EE应用程序

J2ME，micro edtion，主要用于移动设备、嵌入式设备上的java应用程序

Jre【Java Runtime Enviromental】是java运行时环境，那么所谓的java运行时环境，就是为了保证java程序能够运行时，所必备的一基础环境，也就是它只是保证java程序运行的，不能用来开发，而jdk才是用来开发的，所有的Java程序都要在JRE下才能运行。

包括JVM和JAVA核心类库和支持文件。与JDK相比，它不包含开发工具——编译器、调试器和其它工具。

Jre里边包含jvm

Jvm：【Java Virtual Mechinal】因为jre是java运行时环境，java运行靠什么运行，而底层就是依赖于jvm，即java虚拟机，java虚拟机用来加载类文件，java中之所以有跨平台的作用，就是因为我们的jvm

关系：

J2se是基于jdk和jre，

JDK是整个JAVA的核心里边包含了jre，

Jre里边包含jvm

## Java标识符的命名规则

* + - 1. 由字母数字下划线和美元$符号组成
      2. 不能以数字开头
      3. 不能是java关键字
      4. 符合驼峰命名法（首字母大写，对于类名来说，第一个单词的首字母也要大写）
      5. 见名识意
      6. 严格区分大小写

## Java的变量的组成部分

数据类型，变量名，值，存储单元

## 常见的数据类型分类

### 基本数据类型(8个)

* + 1. 数值型（6个）
       1. 整数型（4个）
          1. 字节型 byte 1个字节
          2. 短整型 short 2个字节
          3. 整型 int 4个字节
          4. 长整型 long 8个字节
       2. 浮点型（小数 2个）
          1. float 单精度浮点型 4个字节
          2. double 双精度浮点型 8个字节
    2. 字符型 char 2个字节

布尔类型 boolean 值为：true false 1个字节

### 引用数据类型（3个）

数组（array） 接口（interface） 类（class）

## && 短路与，&逻辑与的区别

* 1. &&短路与
     1. 一旦遇到为结果false的表达式，将不在继续执行运算，直接得到结果false
  2. &逻辑与
     1. 不论中间的表达式结果如何，一定会运算完所有程序表达式才能得到结果

## ||短路或，|逻辑或的区别

* 1. || 短路或
     1. 一旦遇到为结果true的表达式，将不在继续执行运算，直接得到结果true
  2. | 逻辑或
     1. 不论中间的表达式结果如何，一定会运算完所有程序表达式才能得到结果

## While 和do while 的区别

While 先判断循环条件，在执行循环体,若一开始循环条件就不成立，那么将不会执行循环体

do while 先执行循环体，在判断循环条件；不论循环条件是否成立，至少会执行一次循环体

## break和continue的区别

* 1. Break终止循环
  2. Continue结束当前循环开始下一层循环

## 面向对象和面向过程的区别

* 1. 面向过程
     1. 面向事件发生的每一个步骤，一旦某一个环节出现问题，整个过程将无法继续
  2. 面向对象
     1. 面向对象更多的是关注事情发生的结果以及过程中出现的对象，面向对象不关注事件发生的过程，但是过程还是实实在在存在的（万事万物皆对象）

## 重载的概念

* 1. 在同一个类，方法的名称相同，参数的个数，顺序，类型至少有一个不同，那么就构成了方法的重载，方法的重载与返回值类型和修饰符无关

## 构造函数的特点

* 1. 构造方法的基本结构

访问权限修饰符 类名(参数列表){

方法体

}

* 1. 构造方法的作用：实例化对象，初始化创建对象
  2. 每个类中都至少有一个构造方法，当我们没有显示定义构造方法时，系统会默认提供给我们一个无参的构造方法，当我们在类中定义了构造方法后，那么系统将不再提供给我们默认的无参构造
  3. 普通中方法中含有返回值类型，构造函数中没有返回值类型，void也不行
  4. 构造方法的方法名必须和类名相同
  5. 构造方法也可以构成重载

## 成员变量和局部变量的区别

i. 局部变量：定义在方法内部的变量一般来说称之为局部变量作用范围为当前方法

ii. 成员变量/全局变量：定义在方法外部的变量一般来说我们称之为成员变量；作用范围为当前类的所有位置

## 面向对象的三大特性

* 1. 封装 继承 多态

## 封装的步骤

* 1. 私有化属性
  2. 生成公共的get&set方法
  3. 生成有参无参构造

## 继承关键字及其特点：

* 1. 继承的关键字是extends
  2. 继承发生父子类关系中，子类继承了父类，那么就拥有了父类的所有属性和方法
  3. 一个父类可以有多个子类
  4. 一个类只能有一个父类（类的单继承）
  5. 在创建子类对象的时候，默认会调用父类的无参构造，那么若父类中没有无参构造，那么子类就必须手动调用父类的有参构造

## 方法的覆盖/重写override

* 1. 在父子类继承关系中，当父类无法满足子类的需求时，子类需要重写父类的非私有非静态的同名方法，
  2. 方法名称相同，
  3. 参数的个数顺序类型必须完全相同，
  4. 返回值也要必须相同
  5. 子类的访问权限不能小于父类（private default protected public）
  6. 子类不能抛出比父类更多的异常，但可以是父类异常的子异常

## this和super的对比

* 1. this指的是当前对象super指的是父级的对象
  2. 对于方法和属性
     1. this可以调用当前类的属性和方法，格式：this.属性/this.方法()
     2. super调用的是父级的属性和方法，格式：super.属性/super.方法();
  3. 对于构造方法
     1. super可以调用父级的构造方法，一般来说放在子类构造方法的第一行；格式super();
     2. this可以调用当前类的构造方法，一般来说必须放在构造方法的第一行，格式：this()

## 访问权限修饰符及其访问权限

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 本类 | 本包 | 不同包父子类 | 任意位置 |
| private | √ | × | × | × |
| default | √ | √ | × | × |
| protected | √ | √ | √ | × |
| public | √ | √ | √ | √ |

## 抽象类和抽象方法的特点？

* 1. 抽象类
     1. 定义格式：public abstract class 类名
     2. 抽象类中可以有普通方法也可以有抽象方法
     3. 若一个类继承了抽象类那么这个类就必须重写抽象类中的所有抽象方法
     4. 抽象类不能被实例化（抽象类中有构造方法）
  2. 抽象方法
     1. 定义格式：public abstract 返回值类型 方法名(参数列表);
     2. 抽象方法没有方法体
     3. 抽象方法必须存在于抽象类中

## 接口的特点

1. 接口是一种特殊的抽象类
2. 格式：public interface 接口名
3. 接口中所有的方法都是抽象方法；默认都是使用public abstract修饰的
4. 接口中所有的属性都是使用public static final修饰的，也就是说都是静态的常量
5. 接口不能被实例化（接口中没有构造方法）
6. 类与接口之间是实现的关系，即类需要实现接口格式 ： public class 类名 implements 接口名，
7. 类可以实现多个接口（类的多实现），多个接口之间使用，间隔（一个类可以实现多个接口，一个接口可以被多个类实现）
8. 一个类实现了接口，那么就必须重写接口中的所有抽象方法

## final的特点

1. final修饰的变量我们称之为常量；值不能改变，必须赋初始值
2. final修饰的方法不能被重写
3. final修饰的类不能被继承
4. final不能修饰抽象类和抽象方法
5. final不能修饰构造方法

## 多态的体现形式

1. 方法的重写和重载
2. 子类创建父类对象
3. 实现类创建接口对象

## StringBuffer StringBuilder String 区别

1. StringBuffer：线程安全的可变序列，效率低
2. StringBuilder：线程不安全的可变序列，效率高于StringBuffer和String
3. String是一个常量，是不可变的，所以对于每一次+=赋值都会创建一个新的对象StringBuffer和StringBuilder都是可变的，当进行字符串拼接时采用append方法，在原来的基础上进行追加，所以性能比String要高

## List和Set的区别

List有序可重复，Set无序不可重复

## ArrayList 和 linkedList，Vector的区别

* 1. ArrayList 线程不安全；基于数组存储结构，查询速度相对较快，删除和新增相对较慢
  2. LinkedList 线程不安全；基于链式存储结构，查询速度相对较慢，删除和新增相对较快
  3. Vector线程安全

## HashMap和HashTable的区别

* 1. 从继承的类
     1. HashMap继承的是AbstractMap
     2. Hashtable继承的是Dictionary
  2. 安全性
     1. HashMap是线程不安全的
     2. HashTable线程安全的
  3. 空值上来说
     1. HashMap允许键对象和值对象为空值，键对象只允许出现一次null
     2. Hashtable 不允许有空值，否则会出现空指针

## Java中异常的体系结构

1. java 异常是程序运行过程中出现的错误。Java把异常当作对象来处理，并定义一个基类java.lang.Throwable作为所有异常的超类。在Java API中定义了许多异常类,分为两大类，错误Error和异常Exception。其中异常类Exception又分为运行时异常(RuntimeException)和非运行时异常(非runtimeException)，也称之为不检查异常（Unchecked Exception）和检查异常（Checked Exception）。
2. Error与Exception

Error是程序无法处理的错误，比如OutOfMemoryError、ThreadDeath等。这些异常发生时，Java虚拟机（JVM）一般会选择线程终止。

Exception是程序本身可以处理的异常，这种异常分两大类运行时异常和非运行时异常。程序中应当尽可能去处理这些异常。

1. 运行时异常和非运行时异常

**运行时异常:** 都是RuntimeException类及其子类异常： IndexOutOfBoundsException 索引越界异常

ArithmeticException：数学计算异常

NullPointerException：空指针异常

ArrayOutOfBoundsException：数组索引越界异常

ClassNotFoundException：类文件未找到异常

ClassCastException：造型异常（类型转换异常）

这些异常是不检查异常（Unchecked Exception），程序中可以选择捕获处理，也可以不处理。这些异常一般是由程序逻辑错误引起的。

**非运行时异常:**是RuntimeException以外的异常，类型上都属于Exception类及其子类。从程序语法角度讲是必须进行处理的异常，如果不处理，程序就不能编译通过。如：

IOException、文件读写异常

FileNotFoundException：文件未找到异常

EOFException：读写文件尾异常

MalformedURLException：URL格式错误异常

SocketException：Socket异常

SQLException：SQL数据库异常