**终极秘典之JavaSE**

1. **|与||或&与&&之间的区别？**

1、& 和|是位运算符；

2、他们都是逻辑运算符，&& 和||是短路运算，&&如果有一个条件不满足，

则后面不执行，||只要有一个条件满足，后面就不执行；

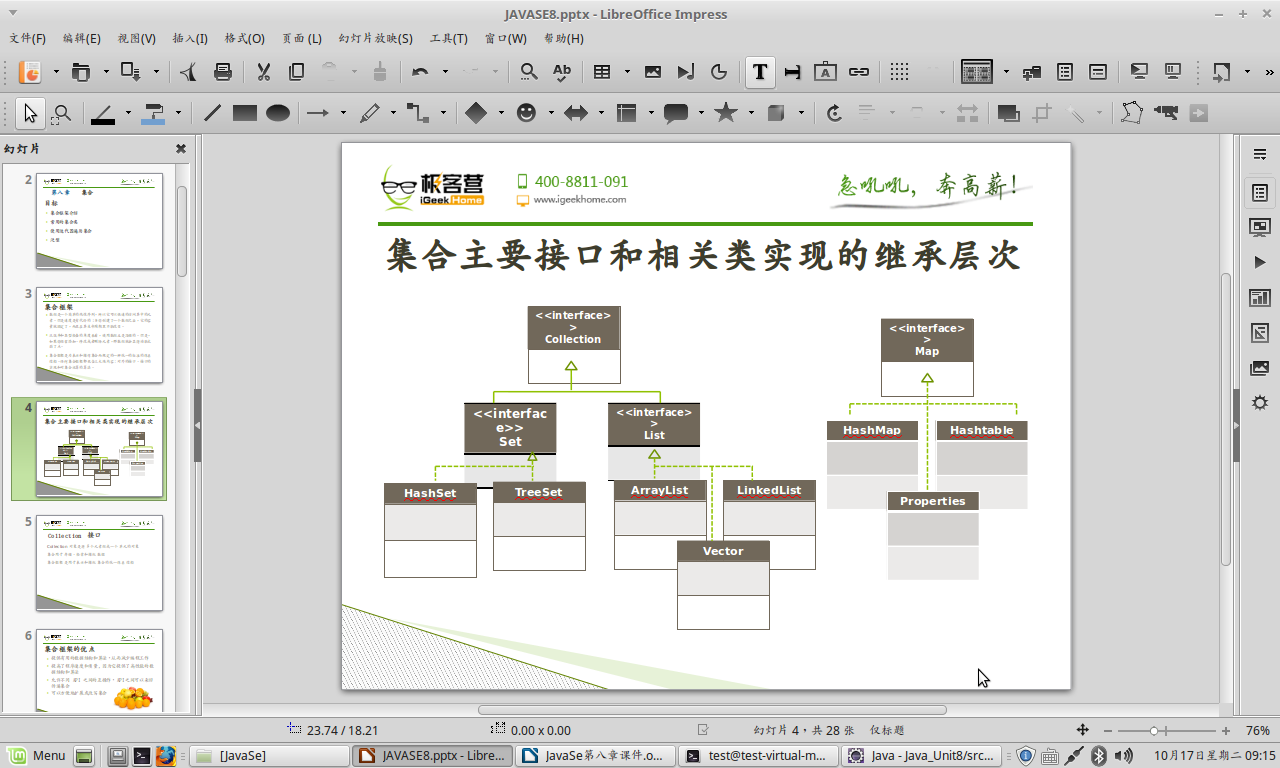
1. **break、continue和return的比较**
2. break结束当前循环；
3. continue结束本次循环，开始下次循环；
4. return结束当前方法；
5. **面向对象与面向过程区别**
6. 面向对象 = 对象 + 消息
7. 面向过程 = 数据 + 算法
8. 面向对象的三大特性：封装 继承 多态
9. Java四大特性：抽象 封装 继承 多态
10. 抽象的概念：把现实中的各个对象（拥有相同属性和行为）抽象成一个Class
11. java计算机世界：先创建Class,然后再new一个一个具体对象
12. **封装**
13. 把属性私有化：用private去修饰
14. 为私有的属性添加get和set方法（在类里，右击选择source-→Generate getters and setters）
15. 为set方法添加控制，保证数据的合理性
16. **值传递问题**
    * 1. 基本数据类型或String这类的包装类型，参数的值不会被改变
      2. 参数是个对象的话，修改对象，对象参数会改变
17. **正则表达式（匹配校验数据格式 ）**
18. \d: 匹配数字（0-9）

\W: 匹配任意单个字符

1. \*:代表任意数量
2. {2}：有2位
3. {3,}：最少有3位，最多不封顶
4. {4,10}：最少有4位，最多有10位
5. 手机号码匹配：

Pattern.*matches*("^1\\d{10}$", phone)

1. **访问修饰符**
2. private只能修饰类自身。
3. 默认（不填写）可以修饰类自身和同包内。
4. protected可以修饰类自身，同包内和子类。
5. Public可以修饰类自身，同包内，子类和所有其他类。
6. **重写和重载**
7. 重写必须继承或实现接口，重写的方法名，参数和返回值都一模一样。
8. 不重写父类的方法，子类对象调用的是父类方法，重写则调用子类重写后的方法。
9. 重载是一个类中，方法名一样，只是参数不一样（参数的个数和类型）。
10. **修饰符Static**
11. static修饰属性（静态属性）
12. static修饰方法（静态方法）
13. static修饰语句块（静态语句块）
14. **修饰符final**
15. final修饰属性（常量）
16. final修饰方法（不能被重写）
17. final修饰类（不能被继承）
18. **修饰符abstract**
19. abstract修饰方法，方法变成抽象方法（必须被重写）
20. 拥有抽象方法的类称为抽象类（必须被继承）
21. 抽象类也可以拥有普通的方法和属性
22. **接口**
23. 继承只能是单个，但是可以支持多个接口
24. 抽象类是abstract class，接口是Interface
25. **处理异常两种方式**
    1. try catch 捕获异常
    2. throws 抛出异常，通常修饰于方法名后，代表调用该方法时可能会触 发异常；
    3. throw 直接抛出一个异常；
26. **throw 与 throws 区别**
27. throws修饰方法（调用这个方法可能会出现异常，需调用者自己捕获处理这个的异常）
28. throw 手动抛出一个异常
29. **final,finalize 与 finally 区别**
30. finalize是Object一个方法，调用它作用是提醒GC去回收这个对象
31. final修饰符 修饰变量成常量 修饰方法不可重写 修饰类不能被继承
32. finally 异常处理中的语法 , 表示 不管try有没有捕获到异常 都会执行的语句块
33. **String和StringBuffer和StringBuilder比较**
    1. 执行速度：StringBuilder＞StringBuffer＞String；
    2. String是字符串常量，其余两个是字符串变量，如果频繁操作字符串优先使用StringBuilder和StringBuffer，效率更高；
    3. StringBuffer是线程安全，而StringBuilder是线程不安全；
34. **实现线程的两种方式**
    1. 继承Thread，重写run方法
    2. 实现Runnable接口，重写run方法
    3. 如何实现线程安全：线程锁synchronized；
35. **排序：**
    1. 选择排序（思想是依次拿出一个值与它所有值相比较，大的放前面）
    2. 冒泡排序（思想是两两相比，大的放前面）
36. **override和overload**
    1. override是重写：必须继承父类，方法名、参数均与父类方法一致，返回值也是相同的；
    2. overload是重载：方法名一致，但参数不一样；
37. **sleep()和wait()比较**
    1. 父类不一样：sleep()来自于Thread，wait()来自于Object；
    2. sleep()没有释放锁，而wait()有释放锁；
    3. sleep()不让出系统资源，而wait()会让出系统资源；
    4. wait()需要用notify()和notifyAll()来唤醒；
38. **集合详解：**



1. **map：键值对 key-value**
   * 1. Hashmap：键值对 key --- value,线程不安全，可以有null值
     2. Hashtable：线程安全，不可以有null值；
     3. TreeMap:有序
2. **collection：**
   1. list：有序、可重复性，有三个实现类

arraylist：使用数组方式存储，查询速度快，增删改慢

linkedlist：链表方式存储，查询慢，增删改快, 存在的意义在于消 除内存碎片，有独有的一些方法;

vector：可以实现可增长(递增)的对象数组,线程安全；

* 1. set：不能重复，有两个实现类

Hashset：由哈希表实现,无序,可以有null值，查询速度快，无下标

Treeset：有序，不可以有null值，查询速度慢；

1. **释放集合**
2. 不用的集合要即使释放掉，集合保存了对象的句柄，ＧＣ不会回收.

list1.clear();

1. **Integer常量池的概念**
   1. Integer常量池的最大值是127.超过则存放到堆里
2. **比较器**
   1. 内部比较器接口（比较的类要实现Comparable接口）：Comparable接口 重写compareTo方法
   2. 外部比较器接口（创建一个独立的比较器类）：Comparator接口 重写compare方法，调用Collections.sort需要把比较器装配进来
3. **泛型**
   1. 集合： 是为了指定集合中存放的数据类型
   2. 接口： 为了适配数据类型
4. **随机数**
5. 随机1~12
   1. ran.nextInt(12) + 1
   2. Math.random()\*12)+1
6. **线程与进程**
7. 进程：每启动一个程序，都会有一个进程
8. 线程：一个进程里可以有多个线程（如边看电影买，边下载）
9. **线程的几种状态**

1. 新建

2. 可运行

3. 等待或阻塞（1.两个线程同时抢一个坑位 2.等待下一个指令 3.超时参数）

4. 终止 （1.正常死亡run运行结束 2.意外死亡抛出错误或异常 3.stop方法但废弃）

1. **线程常用方法**
2. sleep 等待方法
3. join 霸占方法
4. yield 让步方法
5. **内部类的几种写法**

1. 匿名内部类

2. 成员内部类（类里）

3. 局部内部类（方法里）

4. 静态内部类（使用static修饰）

1. **java创建线程的方式有三种**
2. 继承Thread类 实现方法run() 不可以抛异常 无返回值
3. 实现Runnable接口 实现方法run() 不可以抛异常 无返回值
4. 实现Callable<T>接口,接口中要覆盖的方法是 public <T> call() 注意:此方法可以抛异常,而前两种不能 而且此方法可以有返回值
5. **wait-notify 机制，线程锁synchronized**
6. 在 synchronized 方法中才能被调用
7. Wait()等待 ;
8. notify()唤醒 ;
9. notifyAll() 唤醒所有
10. **IO流**
11. **以传输单位分为：**

**字节流**

1.InputStream

2.OutputStream

3.BufferedOutputStream

4.BufferedInputStream

5.DataOutputStream

6.DataInputStream

7.FileInputStream

8.FileOutputStream

9.ObjectOutputStream

10.ObjectInputStream

字符流(抽象父类是 Reader 和 Writer)

1.Writer

2.Reader

3.PrintWriter

4.BufferedReader

5.OutputStreamWriter

6.BufferedWriter

7.InputStreamReader

8.FileWriter

9.FileReader

1. **以流的方向分为：**

输出流

.InputStream

2.FileInputStream

3.DataInputStream

4.BufferedInputStream

5.Reader

6.BufferedReader

7.InputStreamReader

8.FileReader

9.ObjectInputStream

输入流

OutputStream

2.FileOutputStream

3.DataOutputStream

4.BufferedOutputStream

5.Writer

6.BufferedWriter

7.OutputStreamWriter

8.FileWriter

9.ObjectOutputStream

10.PrintWriter

1. **以功能分为：**

节点流（底层流）

1.OutputStream

2.InputStream

3.FileReader

4.FileInputStream

5.FileOutputStream

6.FileWriter

7.Writer

8.Reader

9.PrintWriter

处理流（高层流）

BufferedReader

2.BufferedWriter

3.DataInputStream

4.ObjectInputStream

5.InputStreamReader

6.ObjectOutputStream

7.DataOutputStream

8.OutputStreamWriter

9.BufferedInputStream

10.BufferedOutputStream

1. **JavaBean组件（可重用）**
2. 类必须是public公共的
3. 属性必须是私有的private的，但要提供get和set方法
4. 必须有无参构造
5. **XML** 
   1. 语法跟HTML类似，用来存储和传输数据
   2. XML使用Dom4j解析
6. **反射**
7. 运行时动态获取或加载某个类的属性和方法
8. 对于任意一个对象，都能够调用它的任意一个方法和属性
9. **获取指定Class 的三种方式**
   * 1. 通过全路径名获取 Class.forName
     2. 通过对象获取 stu.getClass
     3. 通过类名获取 Student.class
10. **日志详解**
11. 记录程序的日常运行信息，出问题时好及时定位问题
12. 一般使用log4j记录日志；
13. 日志级别：

info：记录程序逻辑结构

Debug：调试代码；

warning：警报日志；

error：真实报错日志；

1. **socket(套接字) 基于端口号进行长链接通讯，分为服务端(sever)和客户端(client);**

代码：

* + - 1. Sever代码：

ServerSocket ss = **new** ServerSocket(端口号);

Socket socket = ss.accept();

* + - 1. Client代码：

Socket sc = new Socket(IP地址, 端口号);

OutputStream os = sc.getOutputStream();

1. **四种访问修饰符的作用域**

本类 本包 子类 其他类

Public √ √ √ √

Protected √ √ √ X

Delfault √ √ X X

Private √ X X X

1. **接口和抽象类详解：**
2. 抽象类中可以包含抽象方法，也可以有普通方法和属性；
3. 接口只有抽象方法和常量；
4. 抽象类被继承(extends)，接口被实现(implements)；
5. 抽象类用abstract修饰，接口用interface修饰；
6. 共同点；都要重写抽象方法；
7. **==和equals比较**
   * + 1. ==比较基本数据类型时，基本数据类型存放在栈里，比较的是值。

2、 ==比较引用数据类型时，比较的是地址

4、 equals比较引用数据类型，比较的是2个对象的值，显然，当equals为true时，==不一定为true。

1. **Java中如何实现序列化，有什么意义？**
2. 序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。
3. 要实现序列化，需要让一个类实现Serializable接口；
4. **“".equals(user.getName()) 与user.getName().equals("")的区别**
5. **使用**"".equals(user.getName())**不会出现空指针异常**