1、作用域 public,private,protected, 以及不写时的区别

答：区别如下：

作用域 当前类 同一 package 子孙类 其他 package

public √ √ √ √

protected √ √ √ ×

friendly √ √ × ×

private √ × × ×

不写时默认为 friendly

2、ArrayList 和 Vector 的区别 ,HashMap 和 Hashtable 的区别

答：就 ArrayList 与 Vector 主要从二方面来说 .

一.同步性 :Vector 是线程安全的，也就是说是同步的，而 ArrayList 是线程序不安全的，不是同步的

二.数据增长 :当需要增长时 ,Vector 默认增长为原来一培， 而 ArrayList 却是原来的一半

就 HashMap 与 HashTable 主要从三方面来说。

一.历史原因 :Hashtable 是基于陈旧的 Dictionary 类的， HashMap 是 Java 1.2 引进的 Map 接口的一个实现

二.同步性 :Hashtable 是线程安全的，也就是说是同步的，而 HashMap 是线程序不安全的，不是同步的

三.值：只有 HashMap 可以让你将空值作为一个表的条目的 key 或 value

3、char 型变量中能不能存贮一个中文汉字 ?为什么？

答：是能够定义成为一个中文的，因为 java 中以 unicode 编码，一个 char 占 16 个字节，所以放一个中文是没问题的

4、多线程有几种实现方法 ,都是什么 ?同步有几种实现方法 ,都是什么？

答：多线程有两种实现方法，分别是继承 Thread 类与实现 Runnable 接口

同步的实现方面有两种，分别是 synchronized,wait 与 notify

5、继承时候类的执行顺序问题 ,一般都是选择题 ,问你将会打印出什么 ？

答:如下：父类：

```
package test;
public class FatherClass
{
  public FatherClass()
 {
 System.out.println("FatherClass Create");
 }
}
```

子类：

```
package test;
import test.FatherClass;
public class  ChildClass extends FatherClass
{
 public ChildClass()
 {
  System.out.println("ChildClass Create");
 }
 public static void main(String[] args)
 {
  FatherClass fc = new FatherClass();
  ChildClass cc = new ChildClass();
 }
}
```

输出结果：

C:>java test.ChildClass

FatherClass Create

FatherClass Create

ChildClass Create

6、内部类的实现方式 ？

答：示例代码如下：

```
package test;
public class  OuterClass
{
 private class InterClass
 {
  public InterClass()
  {
   System.out.println("InterClass Create");
  }
 }
 public OuterClass()
 {
  InterClass ic = new InterClass();
  System.out.println("OuterClass Create");
 }
 public static void main(String[] args)
 {
  OuterClass oc = new OuterClass();
 }
}
```

输出结果：

C:>java test/OuterClass

InterClass Create

OuterClass Create

7、float　型 float f=3.4　是否正确？

答:不正确。精度不准确　,应该用强制类型转换，如下所示：　float f=(float)3.4

8、介绍 JAVA 中的 Collection FrameWork(　包括如何写自己的数据结构　)?

答：Collection FrameWork　如下：

Collection

├List

│ ├LinkedList

│ ├ArrayList

│ └Vector

│ └Stack

└Set

Map

├Hashtable

├HashMap

└WeakHashMap

9、抽象类与接口？

答：抽象类与接口都用于抽象，但是抽象类 (JAVA 中)可以有自己的部分实现，而接口则完全是一个标识 (同时有多重继承的功能 )。

10、Java 的通信编程，编程题 (或问答 )，用 JAVA SOCKET 编程，读服务器几个字符，再写入本地显示？

答:Server 端程序 :

```java
package test;
import java.net.*;
import java.io.*;

public class Server
{
 private ServerSocket ss;
 private Socket socket;
 private BufferedReader in;
 private PrintWriter out;
 public Server()
 {
 try
 {
  ss=new ServerSocket(10000);
  while(true)
  {
   socket = ss.accept();
   String RemoteIP = socket.getInetAddress().getHostAddress();
   String RemotePort = ":"+socket.getLocalPort();
   System.out.println("A client come in!IP:"+RemoteIP+RemotePort);
   in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
   String line = in.readLine();
   System.out.println("Cleint send is :" + line);
   out = new PrintWriter(socket.getOutputStream(),true);
   out.println("Your Message Received!");
   out.close();
   in.close();
   socket.close();
  }
 }catch (IOException e)
 {
  out.println("wrong");
 }
 }
 public static void main(String[] args)
 {
  new Server();
 }
};
```

Client 端程序 :

```
package test;
import java.io.*;
import java.net.*;

public class Client
{
Socket socket;
BufferedReader in;
PrintWriter out;
public Client()
{
 try
 {
  System.out.println("Try to Connect to 127.0.0.1:10000");
  socket = new Socket("127.0.0.1",10000);
  System.out.println("The Server Connected!");
  System.out.println("Please enter some Character:");
  BufferedReader line = new BufferedReader(new
InputStreamReader(System.in));
  out = new PrintWriter(socket.getOutputStream(),true);
  out.println(line.readLine());
  in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
  System.out.println(in.readLine());
  out.close();
  in.close();
  socket.close();
 }catch(IOException e)
 {
  out.println("Wrong");
 }
}
public static void main(String[] args)
{
 new Client();
}
};
```

11、用 JAVA 实现一种排序， JAVA 类实现序列化的方法 (二种)？ 如在 COLLECTION 框架中，实现比较要实现什么样的接口？

答:用插入法进行排序代码如下：

```
package test;
import java.io.*;
import java.net.*;

public class Client
{
 Socket socket;
 BufferedReader in;
 PrintWriter out;
 public Client()
 {
  try
  {
   System.out.println("Try to Connect to 127.0.0.1:10000");
   socket = new Socket("127.0.0.1",10000);
   System.out.println("The Server Connected!");
   System.out.println("Please enter some Character:");
   BufferedReader line = new BufferedReader(new

InputStreamReader(System.in));
   out = new PrintWriter(socket.getOutputStream(),true);
   out.println(line.readLine());
   in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
   System.out.println(in.readLine());
   out.close();
   in.close();
   socket.close();
  }catch(IOException e)
  {
   out.println("Wrong");
  }
 }
 public static void main(String[] args)
 {
  new Client();
 }
};
```

JAVA 类实现序例化的方法是实现 java.io.Serializable 接口

Collection 框架中实现比较要实现 Comparable 接口和 Comparator 接口

12、编程：编写一个截取字符串的函数，输入为一个字符串和字节数，输出为按字节截取的字符串。　但是要保证汉字不被截半个，如“我　ABC”4，应该截为“我 AB”，输入“我 ABC 汉 DEF”，6，应该输出为“我 ABC”而不是“我 ABC+ 汉的半个”。

答：代码如下：

```
package test;

class  SplitString
{
 String SplitStr;
 int SplitByte;
 public SplitString(String str,int bytes)
 {
  SplitStr=str;
  SplitByte=bytes;
  System.out.println("The String is:"+SplitStr+";SplitBytes="+SplitByte);
 }
 public void SplitIt()
 {
  int loopCount;


 loopCount= (SplitStr.length()%SplitByte==0)?(SplitStr.length()/SplitByte):(SplitStr.length()/Split

 Byte+1);
  System.out.println("Will Split into "+loopCount);
  for (int i=1;i<=loopCount ;i++ )
  {
   if (i==loopCount){


 System.out.println(SplitStr.substring((i-1)*SplitByte,SplitStr.length()));
   } else {


 System.out.println(SplitStr.substring((i-1)*SplitByte,(i*SplitByte)));
   }
  }
 }
 public static void main(String[] args)
 {
  SplitString ss = new SplitString("test中dd文dsaf中男大3443n中国43中国人

 oewldfls=103",4);
  ss.SplitIt();
 }
}
```

喜欢本文的欢迎关注我私信回复“ java 架构”即可免费领取 高并发，分布式，Spring ， MyBatis ， Netty 源码分析和大数据等多个知识点 的架构视频资料！

13、STRING 与 STRINGBUFFER 的区别。

答：STRING 的长度是不可变的， STRINGBUFFER 的长度是可变的。如果你对字符串中的内容经常进行操作，特别是内容要修改时，那么使用 StringBuffer ，如果最后需要 String ，那么使用 StringBuffer 的 toString() 方法

Jsp 方面

1、jsp 有哪些内置对象 ?作用分别是什么 ？

答:JSP共有以下 9 种基本内置组件（可与 ASP 的 6 种内部组件相对应）：

request 用户端请求，此请求会包含来自 GET/POST 请求的参数

response 网页传回用户端的回应

pageContext 网页的属性是在这里管理

session 与请求有关的会话期

application servlet 正在执行的内容

out 用来传送回应的输出

config servlet 的构架部件

page JSP 网页本身

exception　针对错误网页，未捕捉的例外

2、jsp 有哪些动作 ?作用分别是什么 ？

答:JSP共有以下 6 种基本动作

jsp:include ：在页面被请求的时候引入一个文件。

jsp:useBean ：寻找或者实例化一个 JavaBean 。

jsp:setProperty ：设置 JavaBean 的属性。

jsp:getProperty ：输出某个 JavaBean 的属性。

jsp:forward ：把请求转到一个新的页面。

jsp:plugin ：根据浏览器类型为 Java 插件生成 OBJECT 或 EMBED 标记

3、JSP 中动态 INCLUDE 与静态 INCLUDE 的区别？

答：动态 INCLUDE 用 jsp:include 动作实现

<jsp:include page="included.jsp" flush="true" /> 它总是会检查所含文件中的变化，适合用于包含动态页面，并且可以带参数

静态 INCLUDE 用 include 伪码实现 ,定不会检查所含文件的变化， 适用于包含静态页面

<%@ include file="included.htm" %>

4、两种跳转方式分别是什么 ?有什么区别 ？

答：有两种，分别为：

<jsp:include page="included.jsp" flush="true">

<jsp:forward page= "nextpage.jsp"/>

前者页面不会转向 include 所指的页面， 只是显示该页的结果， 主页面还是原来的页面。执行完后还会回来，相当于函数调用。并且可以带参数 .后者完全转向新页面，不会再回来。相当于 go to 语句。

Servlet 方面

1、说一说 Servlet 的生命周期？

答:servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init,service 和 destroy 方法表达。

2、JAVA SERVLET API 中 forward() 与 redirect() 的区别？

答:前者仅是容器中控制权的转向，在客户端浏览器地址栏中不会显示出转向后的地址；后者则是完全的跳转， 浏览器将会得到跳转的地址， 并重新发送请求链接。这样，从浏览器的地址栏中可以看到跳转后的链接地址。所以，前者更加高效，在前者可以满足需要时，尽量使用 forward() 方法，并且，这样也有助于隐藏实际的链接。在有些情况下，比如，需要跳转到一个其它服务器上的资源，则必须使用 sendRedirect() 方法。

3、Servlet 的基本架构

```
public class ServletName extends HttpServlet {
  public void doPost(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
    }
  public void doGet(HttpServletRequest request, HttpServletResponse response) throws
    ServletException, IOException {
    }
}
```

Jdbc、Jdo 方面

1、可能会让你写一段 Jdbc 连 Oracle 的程序,并实现数据查询 .

答:程序如下：

```
package hello.ant;

import java.sql.*;

public class jdbc

{

String dbUrl="jdbc:oracle:thin:@127.0.0.1:1521:orcl";

String theUser="admin";

String thePw="manager";

Connection c=null;

Statement conn;

ResultSet rs=null;

public jdbc()

{

try{
```

```
Class.forName("oracle.jdbc.driver.OracleDriver").newInstance();

c = DriverManager.getConnection(dbUrl,theUser,thePw);

conn=c.createStatement();

}catch(Exception e){

e.printStackTrace();

}

}

public boolean executeUpdate(String sql)

{

try

{

conn.executeUpdate(sql);

return true;

}

catch (SQLException e)

{

e.printStackTrace();

return false;

}

}

public ResultSet executeQuery(String sql)

{
```

```
rs=null;

try

{

rs=conn.executeQuery(sql);

}

catch (SQLException e)

{

e.printStackTrace();

}

return rs;

}

public void close()

{

try

{

conn.close();

c.close();

}

catch (Exception e)

{

e.printStackTrace();

}
```

```
}

public static void main(String[] args)

{

ResultSet rs;

jdbc conn = new jdbc();

rs=conn.executeQuery("select * from test");

try{

while (rs.next())

{

System.out.println(rs.getString("id"));

System.out.println(rs.getString("name"));

}

}catch(Exception e)

{

e.printStackTrace();

}

}

}
```

2、Class.forName　的作用 ?为什么要用 ？

答：调用该访问返回一个以字符串指定类名的类的对象。

3、JDO 是什么？

答:JDO 是 Java 对象持久化的新的规范，为 java data object 的简称 ,也是一个用于存取某种数据仓库中的对象的标准化 API。JDO 提供了透明的对象存储， 因此对开发人员来说， 存储数据对象完全不需要额外的代码 （如 JDBC API 的使用）。这些繁琐的例行工作已经转移到 JDO 产品提供商身上，使开发人员解脱出来，从而集中时间和精力在业务逻辑上。另外， JDO 很灵活，因为它可以在任何数据底层上运行。 JDBC 只是面向关系数据库（ RDBMS)JDO 更通用，提供到任何数据底层的存储功能， 比如关系数据库、 文件、XML 以及对象数据库 （ODBMS ）等等，使得应用可移植性更强。

xml ， EJB 方面：

1、xml 有哪些解析技术 ?区别是什么 ?

答:有 DOM,SAX,STAX 等

DOM: 处理大型文件时其性能下降的非常厉害。 这个问题是由 DOM 的树结构所造成的，这种结构占用的内存较多， 而且 DOM 必须在解析文件之前把整个文档装入内存 ,适合对 XML 的随机访问 SAX:不现于 DOM,SAX 是事件驱动型的 XML 解析方式。它顺序读取 XML 文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理 XML 文件，适合对 XML 的顺序访问

2.EJB 与 JAVA BEAN 的区别？

答:Java Bean 是可复用的组件，对 Java Bean 并没有严格的规范，理论上讲，任何一个 Java 类都可以是一个 Bean。但通常情况下，由于 Java Bean 是被容

器所创建（如 Tomcat) 的，所以 Java Bean 应具有一个无参的构造器，另外，通常 Java Bean 还要实现 Serializable 接口用于实现 Bean 的持久性。Java Bean 实际上相当于微软 COM 模型中的本地进程内 COM 组件，它是不能被跨进程访问的。 Enterprise Java Bean 相当于 DCOM ，即分布式组件。它是基于 Java 的远程方法调用（RMI ）技术的，所以 EJB 可以被远程访问（跨进程、跨计算机）。但 EJB 必须被布署在诸如 Webspere 、WebLogic 这样的容器中， EJB客户从不直接访问真正的 EJB组件，而是通过其容器访问。 EJB 容器是 EJB 组件的代理，EJB 组件由容器所创建和管理。客户通过容器来访问真正的 EJB 组件。

3、 EJB 的基本架构

答:一个 EJB 包括三个部分 ：

Remote Interface 接口的代码

```
package Beans;

import javax.ejb.EJBObject;

import java.rmi.RemoteException;

public interface Add extends EJBObject

{

//some method declare

}
```

Home Interface 接口的代码

```
package Beans;

import java.rmi.RemoteException;
```

```
import jaax.ejb.CreateException;

import javax.ejb.EJBHome;

public interface AddHome extends EJBHome

{

//some method declare

}
```

EJB 类的代码

```
package Beans;

import java.rmi.RemoteException;

import javax.ejb.SessionBean;

import javx.ejb.SessionContext;

public class AddBean Implements SessionBean

{

//some method declare

}
```

J2EE,MVC 方面

1、MVC 的各个部分都有那些技术来实现 ?如何实现？

答:MVC 是 Model －View －Controller 的简写。"Model" 代表的是应用的业务逻辑（通过 JavaBean，EJB 组件实现）， "View" 是应用的表示面（由 JSP 页面产生）， "Controller" 是提供应用的处理过程控制（一般是一个 Servlet ），

通过这种设计模型把应用逻辑, 处理过程和显示逻辑分成不同的组件实现。 这些组件可以进行交互和重用。

2.J2EE 是什么？

答:Je22 是 Sun 公司提出的多层 (multi-diered), 分布式(distributed), 基于组件(component-base) 的企业级应用模型 (enterpriese application model). 在这样的一个应用系统中, 可按照功能划分为不同的组件, 这些组件又可在不同计算机上，并且处于相应的层次 (tier) 中。所属层次包括客户层 (clietn tier) 组件 ,web 层和组件 ,Business 层和组件 ,企业信息系统 (EIS)层。

3.STRUTS 的应用 (如 STRUTS 架构)

答： Struts 是采用 Java Servlet/JavaServer Pages 技术，开发 Web 应用程序的开放源码的 framework 。 采用 Struts 能开发出基于 MVC(Model-View-Controller) 设计模式的应用构架。 Struts 有如下的主要功能：

一.包含一个 controller servlet ，能将用户的请求发送到相应的 Action 对象。

二.JSP自由 tag 库，并且在 controller servlet 中提供关联支持，帮助开发员创建交互式表单应用。

三.提供了一系列实用对象： XML 处理、通过 Java reflection APIs 自动处理 JavaBeans 属性、国际化的提示和消息