1. Socket是基于TCP/IP的传输层协议。   
   Webservice是基于HTTP协议传输数据，http是基于tcp的应用层协议。   
   Webservice采用了基于http的soap协议传输数据。

Socket是传输层的协议，是一种只想着要传输数据的协议，new SOcket对象，有种面向IO流的感觉

Wbservice是一种HTTP协议，传的是封装好的XML对象，有自己的解析机制，有公共性，是XML数据传输，有SOAP钦定的感觉

总结：如果用的是简单的命令、单纯的即时数据通信，用SOcket，如果是应用层的，有富文本等乱七八找的东西，用Webservice通信

(2)Socket接口通过流传输，不支持面向对象。   
Webservice 接口支持面向对象，最终webservice将对象进行序列化后通过流传输。   
Webservice采用soap协议进行通信，不需专门针对数据流的发送和接收进行处理，是一种跨平台的面向对象远程调用技术。

SOcket直接是用流传输，Webservice是序列化之后再用流传输，Webservice因为公用，所以解析什么的包里都做好了

(3)Socket适用于高性能大数据的传输，传输的数据需要手动处理，socket通信的接口协议需要自定义。

   比如：自定义一种字符串拼接的格式，比如自定义的xml数据，自定义麻烦之处在接口调用方和接口服务端需要互相讨论确定接口的协议内容，不方便。   
**缺点**

     程序员需要自己去解析输入、输出流，解析发送和接收的数据。数据传输的格式不固定，需要程序员开发socket接口时自定义接口协议。

**优点**

     如果要传输大数据量，socket可以满足，如果存在大并发使用socket也可以实现，程序用socket灵活性更大，比如可以socket的高并发框架mina开发。

Webservcie由于是遵循标准的soap协议，soap 协议的内容格式固定，soap协议传递的内容是xml数据，由于webservice是基于http的，所以简单理解为soap=http+xml，适用于没有性能要求情况下且数据传输量小，推荐在公开接口上使用webservice，因为soap协议的标准的。

XML+HTTP = WEB = SOAP  
**优点**      jaxws可以通过面向对象开发webservice，程序员不需要解析输入、输出流。   
     由于webservice传输数据使用标准的soap协议(基于http传输xml)，soap协议已经被w3c管理了。   
**缺点**

如果传输大数据量，webservice不适用。如果webservice开发大并发的应用，webservice依靠web容器提高并发数。

WEb容器自己有线程机制，所以并发什么的容器会自己new一些工作线程，容器特么的有自己的线程机制，不该源码控制不了