# Tcp编程

Serversocket

Socket

建立连接 经过3次握手才能确定连接 类似于打电话，连接保持的状态。

输入输出流 发送消息

支持多个用户同时发送消息，把功能放在多线程中

Datainputstream

Dataoutputstream

发送数据，接受数据的

url类 获取网络上的资源的 网站爬虫功能

UDP编程

# 反射技术

有一个类 Class

class 关键字 ，定义类的时候

Class 是类型

Class是类型的一个抽象 类型的类

要操作 不是Class类型 这个类型对应的对象

自己会new

类的加载 jvm去加载类的时候，自动产生这个Class类型的对象的

一个类型对应着一个Class对象

反射机制

代名词

实现类 Class

把我们自己定义类，属性，方法，构造函数 都封装到Class类型中对象

GetFiled获取字段

字段名，修饰名，类型

Filed类

设置值，获取值

Set/get

GetMethod 获取方法

修饰，返回类型 ，参数

Method 类

调用 invoke 唤醒 具体的方法

反射机制

Constractor 处理构造函数的

Newinstance 实例化对象 工厂模式

接口，反射，配置文件，读取配置文件

String className=”User”;

className=”Dog”;

className=”Cat”;

className obj=new className();× 静态处理

Object obj=Class.forName(className).newInstance();√ 动态处理

# 注解功能

元注解

系统 jdk 默认提供的，Sring stringbuffer stringbuilder 自带的

自定义注解

User Dog类我们自定义

什么是注解

Java

J2se j2ee j2me 1.2

Java5se java5ee java5me 1.5

1.7 1.8

// /\*\*/ 注释

注解 也是注释 但是不是给我们程序员看

给java环境看，在运行的时候，可以读取注解，但是不能读注释

Annontation像

一种修饰符一样，应用于包、类型、构造方法、方法、成员变量、参数及本地变量的声明语句中

注解可以用在

注解的主要功能就是配置文件 写在类上 跟这个类一起走（不需要修改的配置）

Properties配置文件 结构简单 key=value

Xml配置文件 结构复杂的

1. **<servlet>**
2. **<servlet-name>**LogOutServlet**</servlet-name>**
3. **<servlet-class>**com.yonyou.mcloud.cas.web.servlet.LogOutServlet**</servlet-class>**
4. **<init-param>**
5. **<param-name>**serverLogoutUrl**</param-name>**
6. **<param-value>**https://dev.yonyou.com:443/sso-server/logout**</param-value>**
7. **</init-param>**
8. **<init-param>**
9. **<param-name>**serverName**</param-name>**
10. **<param-value>**http://10.1.215.40:80/business/**</param-value>**
11. **</init-param>**
12. **</servlet>**
13. **<servlet-mapping>**
14. **<servlet-name>**LogOutServlet**</servlet-name>**
15. **<url-pattern>**/logout**</url-pattern>**
16. **</servlet-mapping>**
17. **<servlet-mapping>**
18. **<servlet-name>**business**</servlet-name>**
19. **<url-pattern>**/**</url-pattern>**
20. **</servlet-mapping>**

## 自定义一个注解

Propertie.load(加载一个配置文件)

读取配置文件

注解定义好了 XX.properties

在类中使用注解，通过反射机制读取注解 Propertie.load（XX.properties）

通过反射赋值 driverName= pro.getProperty(“driverName”)

定义注解

在类中使用注解

读取注解数据，给变量赋值