1. testWeb包下边是一个简单的http服务，启动以后访问localhost:8001/test可以返回“hello world”
2. 重写equals方法

首先定义一个类MyObject，添加getter,setter方法。

重写equals方法，传入一个Object类。

public boolean equals(Object o)

判断，如果两者是同一个对象，则返回true.

if (this == o) return true;

if (!(o instanceof com.xxx.xxx.MyObject)) return false;

如果不是同一个类，则返回false.

return getId() == myObject.getId() &&  
 getName().equals(myObject.getName());

通过每个属性判断两个对象值是否一致，如果一致则返回true

1. hashcode方法

Object的hashcode方法可以重写，返回Objects类中的hash方法。

return Objects.*hash*(getId(), getName());

如果不重写hashcode方法，两个属性值相同的对象hashcode值是不一样的。

Objects中的hash方法返回的是Arrays下的hashCode方法

public static int hash(Object... values) {  
 return Arrays.*hashCode*(values);  
}

Arrays下的hashCode方法中，传入一个Object对象数组。如果数组为null，则返回0.

public static int hashCode(Object a[]) {  
 if (a == null)  
 return 0;  
  
 int result = 1;  
  
 for (Object element : a)  
 result = 31 \* result + (element == null ? 0 : element.hashCode());  
 return result;  
}

如果不为null，则返回一个result的累加。

element.hashCode()

是调用的Object里的hashCode方法。

这里的31选择从网上搜罗到的原话是：

选择数字31是因为它是一个奇质数，如果选择一个偶数会在乘法运算中产生溢出，导致数值信息丢失，因为乘二相当于移位运算。选择质数的优势并不是特别的明显，但这是一个传统。同时，数字31有一个很好的特性，即乘法运算可以被移位和减法运算取代，来获取更好的性能：31 \* i == (i << 5) - i，现代的 Java 虚拟机可以自动的完成这个优化。