数据格式：xml和json

xml：可扩展性标记语言

xml结构的特点:

1.文档声明必须放在第一行

2.标签是自定义的

3.xml文档中包含多个元素

4.元素有开始标签<age>,结束标签</age>,内容组成根据标签的个数分为:

a:双标签元素　　比如<usename>张三</usename>

b.单标签元素　　比如:<age/>

根据元素包含关系分为:

a.父元素

b.子元素

c.根元素:没有父元素的元素

d.叶子元素:没有子元素的元素

e.兄弟元素:同级的元素

5.一个xml文档中只能有一个根元素

6.可以在开始标签中添加多个属性

7.一个元素的属性名不能重复

```
//练习:把以下信息存在xml结构
//雷神 托儿 雷神之锤
//钢铁侠 托尼斯塔斯 盔甲
//美国队长 斯蒂芬 盾牌
```

```xml
<?xml version="1.0" encoding="utf-8" ?>
<!--heroes>hero*3>nickname+name+weapon-->
<accounts>
    <account>
        <nickname>雷神</nickname>
        <name>托儿</name>
        <weapon>雷神之锤</weapon>
    </account>
    <account>
        <nickname>钢铁侠</nickname>
        <name>托尼斯塔斯</name>
        <weapon>盔甲</weapon>
    </account>
    <account>
        <nickname>美国队长</nickname>
        <name>斯蒂芬</name>
        <weapon>盾牌</weapon>
    </account>
</accounts>
```

json:js对象语法

1.没有注释

2.两种结构:{}代表对象,[]代表数组

{}有多个属性,多个属性用逗号隔开

[]中的可以有有多个元素.多个元素用逗号隔开

解析(decode):从xml或json中把数据提取出来

编码(encode):把数据转成xml或json

xml的方法:

1.sax解析:逐行解析

特点:读一行xml内容,解析一行,省系统的资源;xml的格式出现错误,错误之前的内容都可以解析

2.dom解析:全文解析

特点:把整个文档读完后,在解析.占用内存会比较多;xml的格式出现错误,不在进行解析;形成文档的结构模型,使用xpath技术

先在下面写xmlString的xml压缩然后,

异常alt+enter。用替代齐遍历。

```
//解析xml字符串,形成文档模型
try {
    Document document = DocumentHelper.parseText(xmlString);
    //获取文档的根元素
    Element rootElements = document.getRootElement();
    System.out.println(rootElements);
    System.out.println(rootElements.getName());//获取根元素的名字
    System.out.println(rootElements.getText());//获取元素的内容
    System.out.println(rootElements.getStringValue());//不包含表签
    //获取元素的子元素
      rootElements.element("account");
      rootElements.elements("account");
    ArrayList<Hero> accountList = new ArrayList<>();
    List<Element> exceptionList = rootElements.elements( s: "account");
    for (Element accountElement : exceptionList) {
```

在递代器中获取元素的值

```
        String nickname = accountElement.elementText( s: "nickname");//获取子元素的内容
        String name = accountElement.elementTextTrim( s: "name");
        String weapon = accountElement.elementTextTrim( s: "weapon");

        System.out.println(nickname);
        System.out.println(name);
        System.out.println(weapon);

        Hero hero = new Hero(nickname, name, weapon);
        accountList.add(hero);


    }
    System.out.println(accountList);
```

json的解析

```
static void jsonDecode() {
    String jsonString="[{\\\"nickname\\\":\\\"雷神\\\",\\\"name\\\":\\\"托
/       JSON.parseArray();//json解析,最外层是[],使用parseArray
/       JSON.parseObject();//最外层是{},使用parseArray
    List<Hero> heroList = JSON.parseArray(jsonString, Hero.class);
    System.out.println(heroList);

}
```

最后在主方法中调用

```
//xml:dom4j
//json:fast

    xmlDecode();
//xmlEncode();
```

```
//dom4j中使用xpth技术,需要导入jaxen-1.1.1.6jai
try {
    Document document=DocumentHelper.parseText(xmlString);
    //通过xpath技术获取元素
      document.selectNodes();
      document.selectSingleNode()
    //获取英雄的名字
    List<Node> NodeList = document.selectNodes( s: "/accounts/account/name");//      "//hero/na
    for (Node node: NodeList) {
        String text=node.getText();
        System.out.println(text);
    }

} catch (DocumentException e) {
    e.printStackTrace();
}
```