在androd手机中处理xml数据时很常见的事情，通常在不同平台传输数据的时候，我们就可能使用xml，xml是与平台无关的特性，被广泛运用于数据通信中，那么在android中如何解析xml文件数据呢？

通常有三种方式:DOM,SAX,PULL

在这一节中我们使用DOM方式来处理。

DOM方式解析xml是先把xml文档都读到内存中，然后再用DOM API来访问树形结构，并获取数据的，但是这样一来，如果xml文件很大呢？手机CPU处理能力当然不能与PC机器比，因此在处理效率方面就相对差了，当然这是对于其他方式处理xml文档而言。

解析xml文档，当然必须有xml文档文件啦，我自己胡乱弄了一个river，放在assets目录.如下：

<?xml version="1.0" encoding="utf-8"?>

<rivers>

<river name="灵渠" length="605">

<introduction>

灵渠在广西壮族自治区兴安县境内，是世界上最古老的运河之一，有着“世界古代水利建筑明珠”的美誉。灵渠古称秦凿渠、零渠、陡河、兴安运河，于公元前214年凿成通航，距今已2217年，仍然发挥着功用。

</introduction>

<imageurl>

http://imgsrc.baidu.com/baike/pic/item/389aa8fdb7b8322e08244d3c.jpg

</imageurl>

</river>

<river name="胶莱运河" length="200">

<introduction>

胶莱运河南起黄海灵山海口，北抵渤海三山岛，流经现胶南、胶州、平度、高密、昌邑和莱州等，全长200公里，流域面积达5400平方公里，南北贯穿山东半岛，沟通黄渤两海。胶莱运河自平度姚家村东的分水岭南北分流。南流由麻湾口入胶州湾，为南胶莱河，长30公里。北流由海仓口入莱州湾,为北胶莱河,长100余公里。

</introduction>

<imageurl>

http://imgsrc.baidu.com/baike/pic/item/389aa8fdb7b8322e08244d3c.jpg

</imageurl>

</river>

<river name="苏北灌溉总渠" length="168">

<introduction>

位于淮河下游江苏省北部，西起洪泽湖边的高良涧，流经洪泽，青浦、淮安，阜宁、射阳，滨海等六县(区)，东至扁担港口入海的大型人工河道。全长168km。

</introduction>

<imageurl>

http://imgsrc.baidu.com/baike/pic/item/389aa8fdb7b8322e08244d3c.jpg

</imageurl>

</river>

</rivers>

那么如何处理呢？

具体思路是：

\*首先利用DocumentBuilderFactory创建一个DocumentBuilderFactory实例

\*然后利用DocumentBuilderFactory创建DocumentBuilder

\*然后加载XML文档（Document）,

\* 然后获取文档的根结点(Element)，

\* 然后获取根结点中所有子节点的列表（NodeList），

\* 然后使用再获取子节点列表中的需要读取的结点。

当然我们观察节点，我需要用一个River对象来保存数据，抽象出River类

public class River implements Serializable {

private static final long serialVersionUID = 1L;

private String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getLength() {

return length;

}

public void setLength(int length) {

this.length = length;

}

public String getIntroduction() {

return introduction;

}

public void setIntroduction(String introduction) {

this.introduction = introduction;

}

public String getImageurl() {

return imageurl;

}

public void setImageurl(String imageurl) {

this.imageurl = imageurl;

}

private int length;

private String introduction;

private String imageurl;

}

下面我们就开始读取xml文档对象，并添加进List中：代码如下：

我们这里是使用assets中的river.xml文件，那么就需要读取这个xml文件，返回输入流。

读取方法为：inputStream=this.context.getResources().getAssets().open(fileName); 参数是xml文件路径，当然默认的是assets目录为根目录。

然后可以用DocumentBuilder对象的parse方法解析输入流，并返回document对象，然后再遍历doument对象的节点属性。

//获取全部河流数据

/\*\*

\* 参数fileName：为xml文档路径

\*/

public List<River> getRiversFromXml(String fileName){

List<River> rivers=new ArrayList<River>();

DocumentBuilderFactory factory=null;

DocumentBuilder builder=null;

Document document=null;

InputStream inputStream=null;

//首先找到xml文件

factory=DocumentBuilderFactory.newInstance();

try {

//找到xml，并加载文档

builder=factory.newDocumentBuilder();

inputStream=this.context.getResources().getAssets().open(fileName);

document=builder.parse(inputStream);

//找到根Element

Element root=document.getDocumentElement();

NodeList nodes=root.getElementsByTagName(RIVER);

//遍历根节点所有子节点,rivers 下所有river

River river=null;

for(int i=0;i<nodes.getLength();i++){

river=new River();

//获取river元素节点

Element riverElement=(Element)(nodes.item(i));

//获取river中name属性值

river.setName(riverElement.getAttribute(NAME));

river.setLength(Integer.parseInt(riverElement.getAttribute(LENGTH)));

//获取river下introduction标签

Element introduction=(Element)riverElement.getElementsByTagName(INTRODUCTION).item(0);

river.setIntroduction(introduction.getFirstChild().getNodeValue());

Element imageUrl=(Element)riverElement.getElementsByTagName(IMAGEURL).item(0);

river.setImageurl(imageUrl.getFirstChild().getNodeValue());

rivers.add(river);

}

}catch (IOException e){

e.printStackTrace();

} catch (SAXException e) {

e.printStackTrace();

}

catch (ParserConfigurationException e) {

e.printStackTrace();

}finally{

try {

inputStream.close();

} catch (IOException e) {

e.printStackTrace();

}

}

return rivers;

}