7.2.1 Android XML数据解析

Android提供的三种解析XML数据的方案，分别时：SAX，DOM，PULL。

1.XML数据要点介绍

可扩展标记语言，类似于HTML（超文本标记）。更多的时候XML用来存储数据，可以看作一个微型的数据库。比如SharedPreference就是用XML文件来保存配置信息，SQLite的底层也是XML文件。在网络应用方面通常作为信息的载体，通常把数据包装成XML来传递。

文档开始

开始元素(persons)

文本结点(空白文本) 开始元素(person) 属性

文本结点(空白文本) 开始元素(name) 文本结点 结束元素

文本结点(空白文本) 开始元素(age) 文本结点 结束元素

文本结点(空白文本) 结束元素

……

结束元素(persons)

文档结束

<?xml verSion=”1.0” encoding=”UTTF-8”?>

<persons>

<person id = “11”>

<name>Coder-pig</name>

<age>20</age>

</person>

<person id = “13”>

<name>Jay</name>

<age>20</age>

</person>

</persons>

这是简单定义一个存储person对象的xml文件的编码，注意外面的空白区域也是文本结点。

2.三种解析XML文件方法的比较

SAX：对文档进行顺序扫描，扫描到文档（document）开始与结束，元素（element）开始与结束等地方时通知事件处理函数，由事件处理函数做相应的动作，然后继续扫描直至文档结束。解析速度快，占用该内存也少。每解析一类XML，就需要编写新的适合的处理类，用起来麻烦。采用的是流式解析，解析是同步的，读到哪就处理到哪。

DOM：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。写起来简单，但是很消耗内存。假如读取的数据量大，手机内存不够，可能会死机。不建议在Android设备中使用。常用的五个接口与类：Document、Element、Node、NodeList、DOMParser。

DOM是整个文件解析到内存中，供用户需要的结点信息，支持随机访问。

PULL：提供了开始元素和结束元素。当某个元素开始时，可以调用parser.nextText0从XML文档中提取所有字符数据。当解释到文档结束时，自动生成EndDocument。常用接口和类：XmlPullParser、XmlSreializer、XmlPullParserFactory，和SAX差不多，代码实现较为简单，非常适合移动设备。Android系统内置pull解析器，而且Android系统内部默认使用pull来解析XML文件，如果需要在J2EE或者其它地方使用pull修阿婆导入两个包。

3.SAX解析XML数据

SAX解析速度快，内存占用少，非常适合Android设备；解析采用的是事件驱动，不需要解析整个文档，而是再解析过程中判断字符是否符合XML语法的某部分（文件开头结束或者标签开头结束），符合的话就会触发事件（回调方法）,而这些方法都定义在ContentHandler接口中。ContentHandler是一个接口，使用起来不方便，Android提供了一个帮助类DefaultHandler，只需要继承这个类，重写对应的方法。

可重写的方如下：

startDocument()：当读取到文档开始标志时触发，通常在这里完成初始化操作

endDocument()：文档结束部分，在这里完成一些善后工作

startElement(namespaceURI, localName, qName, atts)：参数依次为命名空间、不带命名空间的前缀标签名、带命名空间前缀名的标签、通过atts可以得到所有的属性名与相应的值；SAX一个重要的特点就是流式处理，当遇到标签时，并不会记录以前碰到的标签，也就是说再startElement()方法中，所有知道的信息就是标签的名字和属性，至于标签的嵌套结构，上层标签的名字，是否由子元素等等其它与结构相关的信息都是不得而知的，需要程序来完成，这使得SAX在编程处理上没有DOM来的方便。

endElement(uri, localName, name)：在遇到结束标签时，调用

characters(ch, start, length)：这个方法用来处理在XML文件中读到的内容，第一个参数用于存放文件的内容，后面两个参数是读到的字符串在这个数组中的其实位置和长度，使用new String(ch, start, length)就可以获取内容。

核心代码：

4.DOM解析XML数据

DOM解析时会将XML文件的所有内容以文档树的形式存放到内存中，然后允许使用DOM API遍历XML树，检索需要的数据。使用DOM操作XML的代码看起来比较直观，并且在编码方面比SAX简单。但是因为DOM需要将XML文件的所有内容以文档树方式存放在内存中，所以内存消耗比较大，特别对于运行Android的移动设备来说，因为设备的资源比较宝贵，所以建议采用SAX来解析XML文件，当然如果XML文件内容比较小采用DOM也是可行的。

DOM的API：

DocumentBuilderFactory（解析器工厂类），创建方法：

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder（解析器类），创建方法：通过解析器工厂获得

DocumentBuilder dbBuilder = dbFactory.newDocumentBuilder();

Document（文档树模型）：将要解析的XML文件读入DOM解析器

Document doc = dbBuilder.parse(context.getAssets().open(“person2.xml”));

Document对象代表了一个xml文档的模型树，所有其它Node都以一定的顺序包含在Document对象之内，排列成一个树状结构，以后对XML文档的所有操作都与解析器无关

NodeList（结点列表类）：代表一个包含一个或者多个Node的列表，可看作数组，有以下两个方法： item(index)：返回集合的第index个Node项

getLength()：列表的结点数

Node（结点类）：DOM中最基本的对象，代表文档树中的抽象结点，很少会直接使用，通常时调用它的子对象Element、Attr、Text等。

Element（元素类）：Node最主要的子对象，在元素中可以包含属性，因此有取属性的方法：getAttribute()获得属性值；getTagName()；元素的名称。

Attr（属性类）：代表某个元素的属性，虽然Attr继承自Node接口，但因为Attr时包含在Element中的，但并不能将其看作是Element子对象，因为Attr不是DOM树的一部分

核心代码：DomHelper

5.PULL解析XML数据

除了前面的SAX和DOM解析XML文件外，Android系统内置了PULL解析器用来解析XML文件，比如我们前面学到的SharedPreference就是使用内置pull解析配置文件的。使用和SAX类似，都是采用事件驱动来完成XML的解析。PULL的编码较为简单，只需处理开始与结束事件，通常是用switch语句，根据事件的不同类型匹配不同的处理方式。

流程：

1,获得一个XmlPullParser类的引用

1）通过XML解析工厂获得实例

XmlPullParserFactory factory = XmlPullParserFactory.newInstance();

XmlPullParser parser = factory.newPullParser();

2）直接获取实例

XmlPullParser parser = Xml.newPullParser();

2.为parser解析器对象提供xml流与编码格式

parser.setInput(xml, “UTF-8”);

3.获得事件的类型

int eventType = parser.getEventType();

4.用switch对不同的事件类型进行处理

1）START\_DOCUMENT：开始读文档时触发，在这里完成初始化操作

2）START\_TAG：开始读标签，通过parser的getName方法获得标签名进行比较，使用getAttributeValue(index)可取出属性值。

3）对于文字结点TEXT可使用parse.nextText()获得结点内容

4）END\_TAG：标签结束

5）parser.next()：循环解析下一个元素

核心代码：

使用Pull生成xml数据的流程：

1）创建XmlSerializer（Xml序列化类）实例

XmlSerializer serializer = Xml.newSerializer();

2）为XmlSerizlizer设置输出流与编码格式

serializer.SetOutput(out, “UTF-8”);

3）为XmlSerializer设置xml的编码格式

serializer.startDocument(“UTF-8”, true)

4）设置跟元素

serializer.startTag(null, “persons”);

5）使用for循环遍历perosns集合中的所有元素，同时依次写入标签属性

for(Person p : persons){

serializer.startTag(null, “person”);

serializer.attribute(null, ”id”, p.getId() + “”);

serializer.startTag(null, “name”);

serializer.text(p.getName());

seralizer.endTag(null, “name”);

...

serlizer.endTag(null, “person”);

}

6）设置根完结元素

serializer.endTag(null, “persons”);

7）结束文档编写

serializer.endDocument();

8）调用flush()将内存中的数据写入文件中，关闭输出流

out.flush();

out.close();

核心代码

小结：

介绍了Android中三种常用的XML解析方式:DOM、SAX、PULL。移动端建议用后面两种，而PULL用起来更加简单。