

简介

- XML被设计用来结构化、存储和传输数据。
- XML是纯文本
- XML可自定义标签
- XML是树形结构

语法

规则列表

- 所有 XML 元素都须有**结束标签**
- XML标签对**大小写**敏感
- XML必须正确地**嵌套**
- XML文档必须要有**根元素**
- XML 的属性值须加**双引号**
- XML中空格被保留
- XML 以 LF 存储换行
- XML的转义字符

<	<	小于
>	>	大于
&	&	和号
'	'	单引号
"	"	引号

- XML注释

元素

XML 元素指的是从（且包括）开始标签直到（且包括）结束标签的部分。

XML 元素必须遵循以下命名规则：

- 名称可以含字母、数字以及其他的字符
- 名称不能以数字或者标点符号开始
- 名称不能以字符“xml”（或者 XML、Xml）开始
- 名称不能包含空格

避免使用XML属性而引起的一些问题：

- 属性无法包含多重的值（元素可以）
- 属性无法描述树结构（元素可以）
- 属性不易扩展（为未来的变化）
- 属性难以阅读和维护

下面的三个 XML 文档包含完全相同的信息：

第一个例子中使用了 date 属性：

```

1 <note date="08/08/2008">
2 <to>George</to>
3 <from>John</from>
4 <heading>Reminder</heading>
5 <body>Don't forget the meeting!</body>
6 </note>

```

第二个例子中使用了 date 元素：

```

1 <note>
2 <date>08/08/2008</date>
3 <to>George</to>
4 <from>John</from>
5 <heading>Reminder</heading>
6 <body>Don't forget the meeting!</body>
7 </note>

```

第三个例子中使用了扩展的 date 元素（这是我的最爱）：

```

1 <note>
2 <date>
3   <day>08</day>
4   <month>08</month>
5   <year>2008</year>
6 </date>
7 <to>George</to>
8 <from>John</from>
9 <heading>Reminder</heading>
10 <body>Don't forget the meeting!</body>
11 </note>

```

命名空间

XML 命名空间提供避免元素命名冲突的方法

默认的命名空间（Default Namespaces）

为元素定义默认的命名空间可以让我们省去在所有的子元素中使用前缀的工作。

请使用下面的语法：

```

1 xmlns="namespaceURI"

```

这个 XML 文档携带着某个表格中的信息：

```

1 <table xmlns="http://www.w3.org/TR/html4/">
2   <tr>
3     <td>Apples</td>
4     <td>Bananas</td>
5   </tr>
6 </table>

```

此 XML 文档携带着有关一件家具的信息：

```
1 <table xmlns="http://www.w3school.com.cn/furniture">
2   <name>African Coffee Table</name>
3   <width>80</width>
4   <length>120</length>
5 </table>
```

XML DOM (Document Object Model)

DOM 把 XML 文档视为一种树结构。通过这个 DOM 树，可以访问所有的元素。可以修改它们的内容（文本以及属性），而且可以创建新的元素。元素，以及它们的文本和属性，均被视为节点。

XML 文档中的每个成分都是一个节点，DOM 把XML文档视为一棵节点树 (node-tree)，规定如下：

- 整个文档是一个文档节点(根元素节点)
- 每个 XML 标签是一个元素节点
- 包含在 XML 元素中的文本是文本节点
- 每一个 XML 属性是一个属性节点
- 注释属于注释节点

节点操作：

序号	操作	
1	获取节点	获取元素节点 获取属性节点
2	修改节点	setNodeValue setAttribute
3	删除节点	删除元素节点：removeChild 删除自身 - 删除当前的节点parentNode、removeChild 删除文本节点 removeChild 清空文本节点 nodeValue 根据名称删除属性节点removeAttribute 根据对象删除属性节点removeAttributeNode
4	替换节点	替换元素节点 replaceChild 替换文本节点中的数据 replaceData
5	创建节点	创建新的元素节点：createElement 通过使用 setAttribute 来创建属性 创建新的属性节点：createAttribute 创建文本节点：createTextNode 创建注释节点：createComment
6	添加节点	添加子节点：appendChild 添加父节点：insertBefore insertAfter
7	克隆节点	cloneNode

QT XML

QDomProcessingInstruction

QDomDocument

QDomNode

QDomNodeList

QDomElement

QDomAttr

QDomText

QDomComment

```
1  #include "mainwindow.h"
2
3  #include <QApplication>
4  #include <QtXml>
5  #include <QFile>
6
7
8  void createXmlHead(QString filename,QString workshop)
9  {
10     //xml DOM对象
11     QDomDocument doc;
12
13     //xml文件头
14     QDomProcessingInstruction xmlHead =
15     doc.createProcessingInstruction("xml","version=\"1.0\" encoding=\"UTF-
16     8\"");
17     doc.appendChild(xmlHead);
18
19     //创建根元素节点
20     QDomElement root = doc.createElement(workshop);
21     doc.appendChild(root);
22
23     //xml 文件保存
24     QFile file(filename);
25     if(!file.open(QFile::WriteOnly|QFile::Truncate))
26         return ;
27     QTextStream outStream(&file);
28     doc.save(outStream,4);
29     file.close();
30 }
31
32 void addXml(QString filename,QString cs,QString CameraNum,QString
33 id,QString place,QString clientIP,QString serverIP)
34 {
35     //打开或创建文件
36     QFile file(filename); //相对路径、绝对路径、资源路径都行
37     if(!file.open(QFile::ReadOnly))
38     {
39         qDebug() << "open failed";
40         return;
41     }
42
43     QString err;
44     int line,col;
45     QDomDocument doc;
```

```

43     if(!doc.setContent(&file,true,&err,&line,&col))
44     {
45         qDebug() << "setContent failed : " << err << "line:" <<line
<<"col:"<<col;
46         file.close();
47         return;
48     }
49     file.close();
50
51     QDomNode root = doc.documentElement();
52
53     QDomElement csElement = doc.createElement(cs);
54     root.appendChild(csElement);
55
56     csElement.setAttribute("id",id);
57
58     QDomAttr attr = doc.createAttribute("place");
59     attr.setNodeValue(place);
60     csElement.setAttributeNode(attr);
61
62     attr = doc.createAttribute("CameraNum");
63     attr.setNodeValue(CameraNum);
64     csElement.setAttributeNode(attr);
65
66     QDomElement clientElement = doc.createElement("client");
67     csElement.appendChild(clientElement);
68
69     QDomElement ipElement = doc.createElement("ip");
70     clientElement.appendChild(ipElement);
71
72     QDomText text = doc.createTextNode(clientIP);
73     ipElement.appendChild(text);
74
75     QDomElement serverElement = doc.createElement("server");
76     csElement.appendChild(serverElement);
77
78     ipElement = doc.createElement("ip");
79     serverElement.appendChild(ipElement);
80
81     text = doc.createTextNode(serverIP);
82     ipElement.appendChild(text);
83
84     QDomComment comment = doc.createComment(QString::fromLocal8Bit("图像识别客户端"));
85     csElement.insertBefore(comment,clientElement);
86
87     comment = doc.createComment(QString::fromLocal8Bit("图像识别服务端"));
88     csElement.insertBefore(comment,serverElement);
89
90     //xml 文件保存
91     if(!file.open(QFile::WriteOnly|QFile::Truncate))
92         return ;
93     QTextStream outStream(&file);
94     doc.save(outStream,4);
95     file.close();
96 }
97
98 void readXml(QString filename)

```

```

99 {
100     QFile file(filename);
101     if(!file.open(QFile::ReadOnly))
102     {
103         qDebug() << "open failed";
104         return;
105     }
106
107     QString err;
108     int line,col;
109     QDomDocument doc;
110     if(!doc.setContent(&file,true,&err,&line,&col))
111     {
112         qDebug() << "setContent failed : " << err << "line:" <<line
113         <<"col:"<<col;
114         file.close();
115         return;
116     }
117     file.close();
118     //返回根节点
119     QDomElement root=doc.documentElement();
120     qDebug()<<root.nodeName();
121
122     QDomNode node=root.firstChild(); //获得第一个子节点
123     while(!node.isNull()) //如果节点不空
124     {
125         if(node.isElement()) //如果节点是元素
126         {
127             QDomElement e=node.toElement(); //转换为元素，注意元素和节点是两个数
128             据结构，其实差不多
129             qDebug()<<"      "<<e.tagName()<<" "<<e.attribute("id")<<" "
130             <<e.attribute("place")<<e.attribute("CameraNum"); //打印键值对，tagName和
131             nodeName是一个东西
132             QDomNodeList list=e.childNodes();
133             for(int i=0;i<list.count();i++) //遍历子元素，count和size都可以用，
134             可用于标签数计数
135             {
136                 QDomNode n=list.at(i);
137                 if(node.isElement())
138                     qDebug()<<"      "<<n.nodeName()<<":"<<
139                 <<n.toElement().text();
140             }
141             node=node.nextSibling(); //下一个兄弟节点,nextSiblingElement()是下一个
142             兄弟元素，都差不多
143         }
144     }
145
146     void removeXml(QString filename,QString id)
147     {
148         //打开或创建文件
149         QFile file(filename); //相对路径、绝对路径、资源路径都行
150         if(!file.open(QFile::ReadOnly))
151         {
152             qDebug() << "open failed";

```

```

150         return;
151     }
152
153     QString err;
154     int line,col;
155     QDomDocument doc;
156     if(!doc.setContent(&file,true,&err,&line,&col))
157     {
158         qDebug() << "setContent failed : " << err << "line:" <<line
159         <<"col:"<<col;
160         file.close();
161         return;
162     }
163     file.close();
164
165     QDomNodeList list=doc.elementsByTagName(QString::fromLocal8Bit("宽厚板
166     车间")); //由标签名定位
167     for(int i=0;i<list.count();i++)
168     {
169         QDomNodeList listChild =list.at(i).childNodes();
170         for(int j=0;j<listChild.count();j++)
171         {
172             QDomElement e = listChild.at(j).toElement();
173             qDebug()<<e.tagName()<<e.attribute("id");
174             if(e.attribute("id") == id)
175                 e.parentNode().removeChild(e);
176         }
177     }
178
179     // 保存文件
180     if(!file.open(QFile::WriteOnly|QFile::Truncate))
181         return;
182     QTextStream out_stream(&file);
183     doc.save(out_stream,4); //缩进4格
184     file.close();
185 }
186
187 void updateXml(QString filename)
188 {
189     //更新一个标签项,如果知道xml的结构, 直接定位到那个标签上定点更新
190     //或者用遍历的方法去匹配tagname或者attribut, value来更新
191
192     //打开或创建文件
193     QFile file(filename); //相对路径、绝对路径、资源路径都行
194     if(!file.open(QFile::ReadOnly))
195     {
196         qDebug() << "open failed";
197         return;
198     }
199
200     QString err;
201     int line,col;
202     QDomDocument doc;
203     if(!doc.setContent(&file,true,&err,&line,&col))
204     {
205         qDebug() << "setContent failed : " << err << "line:" <<line
206         <<"col:"<<col;
207         file.close();

```

```

205         return;
206     }
207     file.close();
208
209     QDomElement root=doc.documentElement();
210
211     QDomNodeList list=root.elementsByTagName(QString::fromLocal8Bit("宽厚板
车间"));
212
213     QDomNode node=list.at(list.size()-1).firstChild(); //定位到第三个一级子节
点的子元素
214
215     QDomNode oldnode=node.firstChild(); //标签之间的内容作为节点的子节点出现,当前
是Pride and Prejudice
216     node.firstChild().setNodeValue("Emma");
217
218     QDomNode newnode=node.firstChild();
219
220     node.replaceChild(newnode,oldnode);
221
222     if(!file.open(QFile::WriteOnly|QFile::Truncate))
223         return;
224     //输出到文件
225     QTextStream out_stream(&file);
226     doc.save(out_stream,4); //缩进4格
227     file.close();
228 }
229
230 int main(int argc, char *argv[])
231 {
232     createXmlHead(QString::fromLocal8Bit("远程监
控.xml"),QString::fromLocal8Bit("宽厚板车间"));
233
234     addXml(QString::fromLocal8Bit("远程监
控.xml"),"CS_16","5","1","KHB2#","10.47.0.209","192.168.2.14");
235     addXml(QString::fromLocal8Bit("远程监
控.xml"),"CS_17","5","2","KHB2#","10.47.0.208","192.168.2.14");
236     addXml(QString::fromLocal8Bit("远程监
控.xml"),"CS_13","14","3","KHB1#","10.30.0.214","10.30.0.216");
237     addXml(QString::fromLocal8Bit("远程监
控.xml"),"CS_3500","7","5","KHB1#","10.6.7.101","10.6.7.102");
238     addXml(QString::fromLocal8Bit("远程监
控.xml"),"CS_NEW3500","7","6","KHB1#","10.6.4.25","10.6.7.102");
239
240     readXml(QString::fromLocal8Bit("远程监控.xml"));
241
242     removeXml(QString::fromLocal8Bit("远程监控.xml"),"5");
243
244     readXml(QString::fromLocal8Bit("远程监控.xml"));
245
246     return 0;
247 }
248

```