An XML Primer for Android Developers

Outline

- What is XML?
- The components of XML
- The basic syntax rules
- Elements
- Attributes
- Namespaces

What is XML?

- eXtensible Markup Language
- A simple text-based format for representing structured information
  - Documents, data, configurations, transactions, etc.
- Developed by the W3C (World Wide Web Consortium, [http://www.w3.org](http://www.w3.org))
  - Derived from SGML. Simpler.
  - Similar syntax to HTML. More strict. No predefined tags.
- Used extensively in Android for resources and configurations, etc.
Anatomy of an XML Document
– String Resources strings.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">HelloAndroid</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
</resources>
```

Anatomy of an XML Document
– String Resources strings.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">HelloAndroid</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
</resources>
```
Anatomy of an XML Document
– String Resources strings.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app_name">HelloAndroid</string>
  <string name="hello_world">Hello world!</string>
  <string name="action_settings">Settings</string>
</resources>
```

A container element

A child element

The root element

The element content

A container

The start-tag

The end-tag

The content
XML Document Tree

- An XML document forms a tree

![XML Document Tree Diagram]

Basic Syntax Rules of XML

- XML declaration must be at the beginning
  - If not present, default to XML 1.0
- Names are case-sensitive
- White spaces are preserved
- Comments
  ```
  <!-- This is a comment -->
  ``
  - But
  ```
  <!-- This is an -- illegal -- comment -->
  ```

Rules for Tag Names

- May contain letters, digits, and other characters
  - No spaces
  - Must start with a letter
  - Cannot start with the prefix xml (or XML, or Xml, etc.)
- Best naming practices
  - Use simple descriptive names
  - Avoid: - : .

Entity References

- Some characters has special meanings in XML
  - `<` & `>`
  - Illegal for regular contents
- Use entity references for these characters

<table>
<thead>
<tr>
<th>Entity</th>
<th>Character</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&amp;lt;</code></td>
<td><code>&lt;</code></td>
<td>less than</td>
</tr>
<tr>
<td><code>&amp;gt;</code></td>
<td><code>&gt;</code></td>
<td>greater than</td>
</tr>
<tr>
<td><code>&amp;amp;</code></td>
<td><code>&amp;</code></td>
<td>ampersand</td>
</tr>
<tr>
<td><code>&amp;apos;</code></td>
<td><code>'</code></td>
<td>apostrophe, single quote</td>
</tr>
<tr>
<td><code>&amp;quot;</code></td>
<td><code>&quot;</code></td>
<td>double quote</td>
</tr>
</tbody>
</table>
Elements

- May contain other elements.
- End-tag, or closing-tag, is required for all elements

```
<string _ > ... </string>
```
- Empty elements. An element with no contents.

```
<string></string>
```
- Can be abbreviated as

```
<string/>
```

Attributes

- Elements may have attributes
  - Specified in the element start-tag
  - Each attribute has a name and a value

```
name = "value"
<string name="app_name">HelloAndroid</string>
```
- Attribute name must be unique within the element
- Attribute values must be enclosed by quotes
  - Either single (‘…’) or double (“…” ) quotes
  - Values are treated as simple strings

Well-Formed XML Documents

- A document must have a single root element
- All elements must have matching start-tag and end-tag
- All elements must be properly nested
  - Overlapping elements are not allowed
  - Legal:

```
<event> ... <location> ... </location> ... </event>
```
  - Illegal:

```
<event> ... <location> ... </event> ... </location>
```
- All attributes are quoted

Namespaces

- A mechanism to avoid name conflicts
  - Using a prefix: prefix:name

```
<root>
  <my:table> ... </my:table>
  <your:table> ... </your:table>
</root>
```

- The xmlns attribute: xmlns:prefix=uri

```
<root xmlns:my="http://www.myinc.com/"
      xmlns:your="http://www.yourcorp.com/" >
  <my:table> ... </my:table>
  <your:table> ... </your:table>
</root>
```
Using Namespace
– Android App Manifest

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.depaul.csc472.helloandroid">
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MyActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>

Use of Namespaces in Android

- All Android SDK defined names are in the namespace named android
  xmlns:android="http://schemas.android.com/apk/res/android"
- Conventions for XML files in Android projects
  - Namespaces are absent for tag names
    - Assuming only Android SDK defined tags are used
  - The android prefix is used for all Android SDK defined attributes
    - Non Android SDK defined attributes may also be present
    - Attributes not in the android namespace are ignored by Android

Next …

- Android app fundamentals
- Android resources in XML