တတိယခြေလှမ်း

မင်းလွင် XMLဆီသို့ခြေလှမ်းများ

အခြေခံ XSL

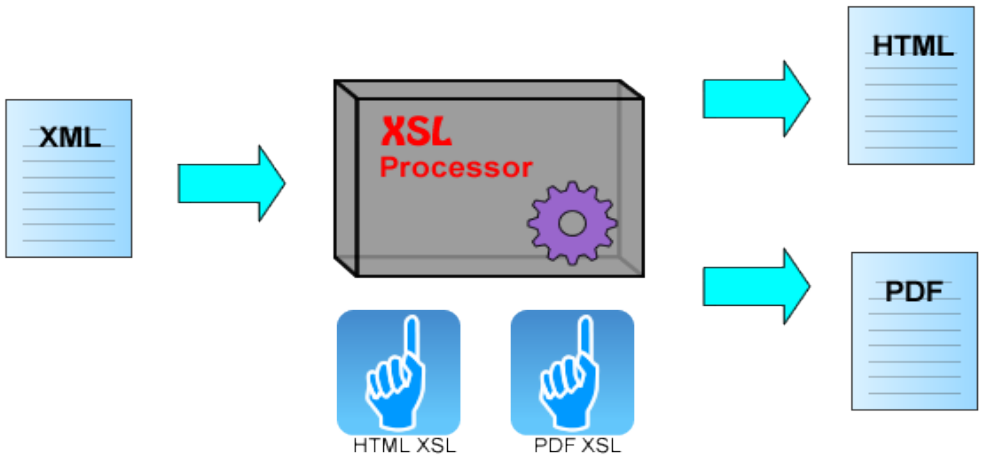
3. တတိယခြေလှမ်း အခြေခံ XSL

XML ရဲ့ Tag များသည် အချက်အလက်တွေရဲ့အဓိပ္ပါယ်များနှင့်၊ ထူးခြားချက်များကိုဖော်ပြပေးနိုင်ပါသည်။ ယခင်အခန်းများတွင် ဖော်ပြပြီးခဲ့သလို XML သည်အချက်အလက်များကို ဖော်ပြရန်အတွက် Tag များကို လွတ်လပ်စွာ အသုံးပြုနိုင်ပေ၏။ ထို့အတွက် HTML ကဲ့သို့ ပိုင်ဆိုင်တဲ့Tagတွေအသီးသီးကို မည်သို့ဖော်ပြမည် ဆိုသည်ကို ကြိုတင်သတ်မှတ်ထားခြင်းမရှိပေ။သို့ရာတွင် XML အချက်အလက်များကို ပြင်ဆင်ပြီးလုပစွာ ပုံဖော်လိုကြမည် မဟုတ်ပေလော။ XML ရဲ့ အချက်အလက်များကို ဘယ်လိုစည်းမျဉ်းဖြင့် ပုံစံဖော်မည်ဆိုသည့် လုပ်ဆောင်ချက်သည် လိုအပ်လာပေသည်။ ထိုတာဝန်များကို ထမ်းဆောင်ပေးနေသည်မှာ XSL ဖြစ်ပေသည်။ XSL ဆိုသည်မှာ Extensible Style sheet Language ရဲ့အတိုကောက်အခေါ်အဝေါ် ဖြစ်ပြီး၊ XML စာပိုဒ်များ၏ Style ကိုသတ်မှတ်ပေးနိုင်သော ပညာရပ်တစ်ခု ဖြစ်၏။

XSL သည် XML ရဲ့တည်ဆောက်ပုံကို ပြောင်းလည်းပေးသော XSLT (XSL Transformations)၊ XML စာပိုဒ်များ၏ အစိတ်အပိုင်းအတိအကျကို ရှာဖွေနိုင်သော XPath (XML Path Language) နှင့်၊ အမြင်ပိုင်း ဆိုင်ရာ Style ကိုဖော်ပြပေးနိုင်သော XSLFO (XSL Formatting Objects) အစရှိသည့် နည်းပညာရပ်များဖြင့် ဖွဲ့စည်းထားပါသည်။ XSL၏XSLFO သည် HTML၏CSS ကဲ့သို့ နေရာတွင်တည်ရှိပေသည်။ ဤတတိယခြေလှမ်းတွင် XSLT ကိုအသုံးပြု၍၊ XML အချက်အလက်များကို Web Browser များတွင် ဖော်ပြနည်း များကို တင်ပြသွားမည် ဖြစ်သည်။ သို့ရာတွင် XSLFO မှာ CSSဖြင့်အစားထိုးအသုံးပြုနိုင်သဖြင့် ဤအခန်းတွင် အကျဉ်းချုံး ဖော်ပြသွားမည် ဖြစ်သည်။

3.1. XSL ရဲ့အခြေခံသဘောတရား

XSL သည် XML စာပိုဒ်များကို XSL အသုံးပြုတဲ့File အပေါ်မှုတည်ပြီး XSL processor က တည်ဆောက်ပုံကို ပြောင်းလည်းပေးခြင်းအားဖြင့် အခြားသောဖွဲ့စည်းပုံအချက်အလက်များ အဖြစ်ပြောင်းလည်းပေးနိုင်ပါသည်။ ဥပမာ အားဖြင့် XML စာပိုဒ်အား HTML သုံးXSL ကိုအသုံးပြုပါက HTML အချက်အလက်အဖြစ် ပြောင်းလည်းပေးနိုင်သလို၊ PDF သုံးXSL ကိုအသုံးပြုပါက PDF အချက်အလက်များအဖြစ် ပြောင်းလည်းပေးနိုင်ပါသည်။



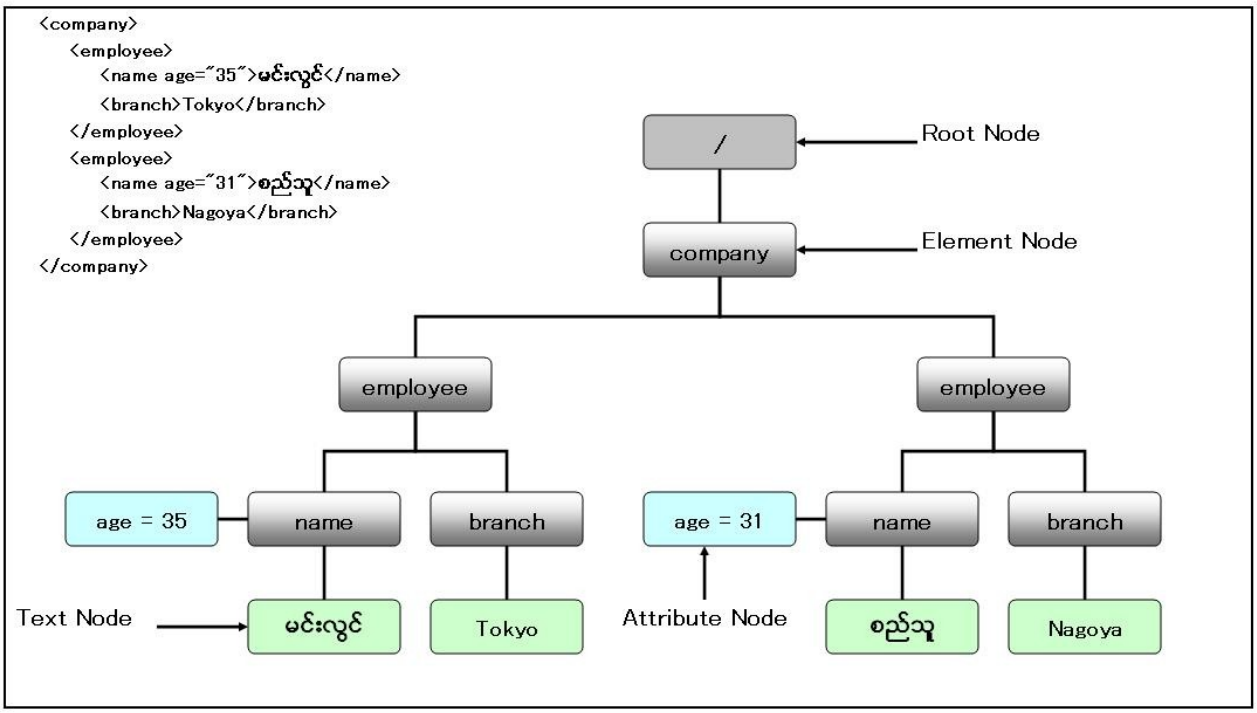
XSL Processor သည်အတော်များများသော Internet Explore တွင် ပါရှိပါသဖြင့်၊ ဤအခန်းတွင်ဖော်ပြသော နမူနာများကို Internet Explore ကိုအသုံးပြု၍ လက်တွေ့လေ့လာသွားမည်ဖြစ်သည်။

3.2. XPath

XPath သည်XSL မှ XML စာပိုဒ်အတွင်းမှ အချက်အလက်များ၏ နေရာအတိအကျကို ဆက်သွယ်ရာတွင်

အသုံးပြုရာတွင် အသုံးပြုသောစနစ်တစ်ခုဖြစ်၏။ XPath မပါဝင်ပဲ XSL ကိုအသုံးပြုနိုင်မည် မဟုတ်ပေ။XPath သည်လည်း XML ကဲ့သို့ပင် သစ်ပင်ဖွဲ့စည်းပုံဖြင့် တည်ဆောက်ထားပါသည်။ XMLနှင့် ကွာခြားသည်မှာ Element အဆင့်မဟုတ်ပဲ၊ Node အဆင့်ဖြင့် သစ်ပင်ဖွဲ့စည်းပုံကို ဖွဲ့စည်းထားခြင်းဖြစ်သည်။

Diagram 3.2.1



3.2.1 Node အမျိုးအစားများ

Xpath တွင်XMLဏ် Element, Attributeနှင့်တန်ဖိုးများကို သီးခြားNodeတွေအဖြစ် သတ်မှတ်ထားပေသည်။ Node အမျိုးအစားများကို အောက်ပါအတိုင်းဂမျိုးခွဲခြား ထားပါသည်။

- Root Node

  XML စာပိုဒ်များတွင် မရှိမဖြစ်လိုအပ်သော တစ်ခုတည်းသော Nodeဖြစ်ပြီး၊ သစ်ပင်ဖွဲ့စည်းပုံ ဏ်ပင်မနေရာတွင်တည်ရှိသော Nodeဖြစ်ပါသည်။ အမြဲတမ်း / ဟုဖော်ပြလေ့ရှိပြီး၊ Node တွေအားလုံးဏ် ပင်မြစ်ဖြစ်ပေသည်။ Root Nodeဏ်အောက်တွင် အမြဲတမ်းRoot Element တစ်ခုတည်ရှိပေမယ့် ထိုElementနှင့် Root Node တို့သည် သီးခြားNode များဖြစ်ကြ ပါသည်။

- Element Node

  XML စာပိုဒ်များ၏ Element အမည်တွင် နေရာကျပေလိမ့်မည်။ အထက်ပါဥပမာတွင် company, employee, name, branch အစရှိသည့် Node များသည် Element Node များ ဖြစ်ကြပါသည်။

- Attribute Node

  XML စာပိုဒ်များ၏ Attribute များ၏နေရာတွင် တည်ရှိပေသည်။ အထက်ပါဥပမာတွင် Age သည် Attribute Nodeဖြစ်၏။

- Text Node

  XML စာပိုဒ်များ၏ Text Value များသည် Text Node များဖြစ်ကြ၏။

3.2.2 Node တစ်ခု၏တည်နေရာ

Xpath တွင်Node တစ်ခုရဲ့ အချက်အလက်ကိုဆက်သွယ်အသုံးပြုနိုင်ရန် ထို Node ရဲ့ တည်နေရာ ကို ဖော်ပြပေးဖို့လိုအပ်ပါသည်။ Node တစ်ခု၏နေရာကိုဖော်ပြရာတွင် Node တစ်ခုနှင့် တစ်ခု အကြား / ကိုကြားခံ၍ ဖော်ပြရပါမည်။ ထိုသို့ဖော်ပြချက်ကို နေရာဖော်ပြချက်(Location Path) ဟု ခေါ်ပါသည်။ Node တစ်ခု၏ တည်နေရာကိုဖော်ပြပုံမှာ ၂မျိုးခွဲခြားနိုင်ပါသည်။ XML စာပိုဒ် တစ်ခုလုံး၏ နေရာအတိအကျကိုဖော်ပြပေးသော အတိအကျနေရာ(Absolute Location Path)နှင့်၊ လက်ရှိNode မှ၊ ရောက်ရှိလိုသော Node ကိုညွှန်ပြသော နှိုင်းယှဉ်နေရာ (Relative Location Path)ဟူ၍ရှိပေသည်။

- အတိအကျနေရာ (Absolute Location Path)

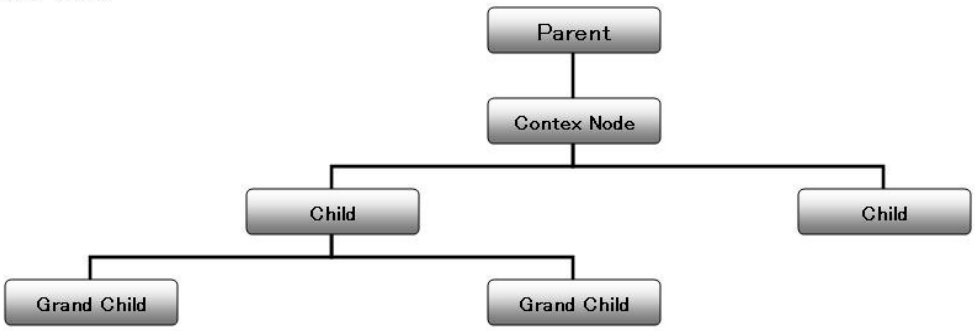
  Root Nodeနေရာမှ ဆက်သွယ်လိုသော Node အထိလမ်းကြောင်းကို အတိအကျနေရာဟု ခေါ်ပါသည်။ Diagram 3.2.1 တွင် name Node၏အတိအကျနေရာ ဖော်ပြချက်မှာအောက်ပါ အတိုင်းဖြစ်သည်။

```
/company/employee/name
```

- နှိုင်းယှဉ်နေရာ (Relative Location Path)

Node တစ်ခုမှ အခြားသော Node တစ်ခု၏ နေရာကိုနှိုင်းယှဉ်၍ ဖော်ပြရာတွင်အသုံးပြုပါသည်။ Node၏အပေါ်တွင်ရှိသော Node ကို Parent Node ဟုခေါ်ပြီး၊ အောက်တွင်ရှိသော Node ကို Child Node ဟု ခေါ်ပါသည်။ သား၏သား Node များသည် Grand Child Node များဖြစ်ကြပါသည်။ ဤကဲ့သို့အခေါ်အဝေါ်များကို XPath Axes ဟုခေါ်ပါသည်။

Diagram 3.2.2



အမည်	အသုံးပြုပုံများ
ancestor	ကိုယ့်၏အထက်မှNode များ
ancestor-or-self	ကိုယ်နှင့်ကိုယ့်၏အထက်မှNode များ
attribute	ကိုယ်ပိုင်ဆိုင်သောAttribute Node များအားလုံး
child	ကိုယ်၏သားအဆင့်ရှိChild Node များအားလုံး
descendant	ကိုယ်၏အောက်မှNode များ
descendant-or-self	ကိုယ်နှင့်ကိုယ်၏အောက်မှNode များ
following	ကိုယ်၏အနောက်မှရှိသောNode များအားလုံး
following-sibling	ကိုယ်၏အနောက်မှမိဘတူညီသော ညီအကိုNode များအားလုံး
parent	ကိုယ်၏အထက်တွင်ရှိသောမိဘ Node
preceding	ကိုယ်၏အရှေ့မှ Node များအားလုံး
Preceding-sibling	ကိုယ့်ရှေ့မှမိဘတူညီသော ညီအကိုNode များအားလုံး
self	ကိုယ်ကိုတိုင်

တဖန် XPath တွင် အတိုကောက်စာလုံးများကိုလည်း အသုံးပြုနိုင်ပေသည်။ အသုံးပြုနိုင်သော အတိုကောက်စာလုံးများမှာ အောက်ပါအတိုင်းဖြစ်ပါသည်။

အတိုကောက်	သက်မှတ်နိုင်သော Node များ
Node name	
/	root
//	descendant-or-self
.	self
..	parent
@	Attribute

Diagram3.2.1၏ ဥပမာကိုကြည့်ပါ။ လက်ရှိNode သည် companyဖြစ်မည်ဆိုလျှင် Name Node ကိုဖော်ပြရာတွင် အောက်ပါအတိုင်းရေးသားနိုင်ပါသည်။

```
employee/name
   သို့မဟုတ်
//name
```

- အမည်တူNode များ

  အကယ်၍လက်ရှိNodeဏ်အောက်တွင် အမည်တူNode များတည်ရှိပါက အောက်ပါအတိုင်း ဖော်ပြရပါမည်။

```
Node ၏အမည် [နံပါတ်]
```

Diagram3.2.1၏ company node တွင် child nodeနှစ်ခုရှိပါသည်။ ဥပမာအားဖြင့် မင်းလွင် အမည်ရှိသော Node သည် employee [1] ဖြစ်၍၊ စည်သူဟုအမည် node ကိုပိုင်ဆိုင်သော node သည် employee [2]ဖြစ်သည်။

အခြားသော ပရိုဂရမ်ဘာသာရပ်များကဲ့သို့ 0 မှအစမပြုသည်မှာ သတိထားစရာအချက် တစ်ခုဖြစ်သည်။

3.3. XSLT (XSL Transformation)

XML စာပိုဒ်တစ်ခု၏ အချက်အလက်များကို ဘယ်လိုပုံစံဖြင့်ဖော်ပြပါမည်ဆိုတဲ့ အချက်အလက်များကို ရေးသားထားခြင်းကို Style Sheet ဟုခေါ်ပါသည်။ ထိုStyle Sheet၏ File Type သည် .xsl ဟုရေးသားဖို့လိုအပ်ပါသည်။ သို့ရာတွင် Style Sheet ကိုယ်တိုင်လည်း XMLရေးသားပုံအတိုင်း ရေးသားထားခြင်းဖြစ်ပါသည်။ ဦးစွာလက်တွေ့ရေးသားကြီးပြီး ဘယ်လိုရေးသားရသည်ကို ရှင်းလင်းသွားပါမည်။

book.xml

```
1   <?xml version="1.0" encoding="UTF-8" ?>
2   <?xml-stylesheet href="book.xsl" type="text/xsl"?>
3   <book-list>
4           <book>
5                   <name>Head First Java</name>
6                   <price>$ 25.00</price>
7           </book>
8           <book>
9                   <name>Effective Java</name>
10                  <price>$ 42.09</price>
11          </book>
12          <book>
13                  <name>Head First Design Patterns</name>
14                  <price>$ 29.00</price>
15          </book>
16  </book-list>
```

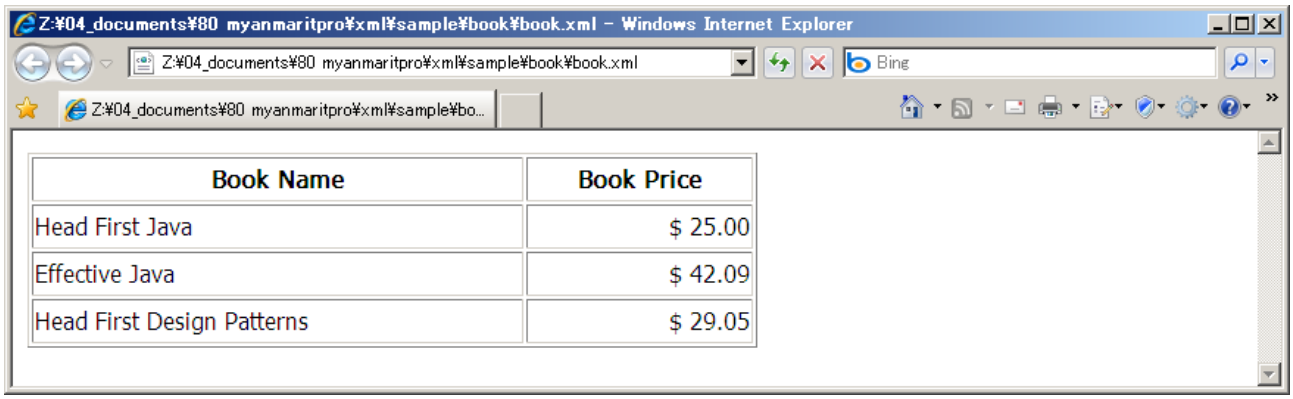
အထက်ပါ book.xml တွင်အသုံးပြုသော Style Sheet သည်အောက်ပါ book.xslဖြစ်၏။

book.xsl

```
1   <?xml version="1.0" encoding="UTF-8" ?>
2   <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
3   <xsl:template match="/">
4           <html>
5           <body>
6           <table border="1">
7                   <tr><th>Book Name</th><th>Book Price</th></tr>
8                   <xsl:apply-templates select="booklist/book" />
9           </table>
10          </body></html>
11  </xsl:template>
12  <xsl:template match="book">
13          <tr>
```

```
14        <td><xsl:value-of select="name" /></td>
15        <td align="right"><xsl:value-of select="price" /></td>
16        </tr>
17    </xsl:template>
18    </xsl:stylesheet>
```

အထက်ပါ book.xml ကို Internet Exploreဖြင့် ဖွင့်ကြည့်မည်ဆိုလျှင် အောက်ပါအတိုင်းတွေ့မြင်ရပေမည်။



3.3.1 Style Sheet ကိုထည့်သွင်းအသုံးချခြင်းအောက်ပါအတိုင်းတွေ့မြင်ရပေမည်။

အထက်ပါ book.xml၏ ဒုတိယမြောက်စာကြောင်းသည် XML စာပိုဒ်အတွင်းတွင် XSL ကိုအသုံးပြုပါမည် ဟုသတ်မှတ်ချက်ပေးနေခြင်းဖြစ်ပါသည်။ ထိုစာကြောင်းကိုရေးသားခြင်းအားဖြင့် XML စာပိုဒ်တွင် အသုံးပြုမည့် Style Sheet ကိုသတ်မှတ်ပြီး ဖြစ်စေပါသည်။ ရေးသားပုံမှာအောက်ပါအတိုင်းဖြစ်ပါသည်။

```
<?xml-stylesheet href="book.xsl" type="text/xsl" ?>
```

3.3.2 Style Sheet အတွင်းမှရေးသားချက်များ

တဖန် Style Sheet ဖြစ်တဲ့ book.xsl ကိုလေ့လာကြည့်ကြပါဆို့။ xsl သည်လည်း XML စာပိုဒ်တစ်မျိုးဖြစ်သည့် အတွက် ပဌမစာကြောင်းသည် XML Declaration ဖြစ်ပါသည်။ ဒုတိယစာကြောင်းမှစ၍ XML Instance ဖြစ်ပြီး stylesheet Tagဖြင့် အစပြုထားပေသည်။

Namespace Uri အဖြစ် http://www.w3.org/1999/XSL/Transform ကိုအသုံးပြုပြီး၊ prefix အဖြစ် xsl ကို အသုံးပြုထားပါသည်။ ထိုစာကြောင်းသည် XSLT ကိုရေသားရာတွင် မရေးမဖြစ်ရေးသားရမည်ဖြစ်၏။ တနည်း ဆိုသော် Stylesheet Element သည် XSLT၏ Root Element အဖြစ် မရှိမဖြစ်လိုအပ်ပါသည်။ ထိုမှတဆင့် ရေးသားလိုသော Style ကို Child Elements များအတွင်း ရေးသားရမည်ဖြစ်သည်။

1. Template Rule

   စာကြောင်း နံပါတ်၃မှ ၁၇အထိသည် Template စည်းမျဉ်း များဖြစ်ကြ၏။ Template စည်းမျဉ်း ဆိုသည်မှာ XML၏အပိုင်းတစ်ခုကို ဘယ်လိုပုံစံပြောင်းရမည်ဟု သက်မှတ်ထားသော စည်းကမ်းများ

ဖြစ်ကြ၏။ စာကြောင်းနံပါတ်၃သည် root node ရဲ့ဖော်ပြပုံပုံစံကို သတ်မှတ်နေ၏။ တဖန်စကြောင်း ၁၂မှစ၍ book node၏ ဖော်ပြပုံဖြစ်၏။ Template Rule Tag၏ရေးသားပုံမှာ အောက်ပါအတိုင်းဖြစ်၏။

```
<xsl:template match="/">          <------  root node
   …
<xsl:template match="book">       <------  single node
```

Template Tag၏ match attribute သည် လက်ရှိTemplate တွင်အသုံးပြုမည့် node ကို Xpathရေးသားပုံ ဖြင့် သတ်မှတ်ရပါမည်။ book.xsl တွင် root node နှင့် book node အသီးသီးကို စာကြောင်း၃နှင့်၁၂တွင် သတ်မှတ်ထားပါသည်။

2. Template Rule ကိုအသုံးချခြင်း

book.xsl၏ စာကြောင်းနံပါတ်၈ကို ကြည့်ပါ။ ထိုစာကြောင်းတွင် booklist/book Element အတွက် Template Rule ကို အသုံးပြုနေပါသည်။ ဤစာကြောင်းသည် Root Node ၏ Template စည်းမျဉ်း အတွင်းရေးသားထားပါသဖြင့် Root Node ကို Context Node အဖြစ်အသုံးပြုပြီး အသုံးပြုမည့် Node များ ကို ရည်ညွှန်း၍ဖော်ပြရပါသည်။

```
<xsl:apply-templates select="booklist/book">
```

Book Elementl၏ Template စည်းမျဉ်းများသည် စာကြောင်း၁၂မှ ၁၇အထိရေးသားထားပြီး XSLT က HTML အဖြစ် ပုံစံပြောင်းလိုက်သောအခါ အောက်ပါအတိုင်းပြောင်းလည်းသွားမည် ဖြစ်သည်။

```
1  <xsl:template match="/">
2          <html>
3          <body>
4          <table border="1">
5                  <tr><th>Book Name</th><th>Book Price</th></tr>
6  <!-- ဤနေရာမှ  Book Node၏ Template စည်းမျဉ်းတွင်သတ်မှတ်ထားသည့်အတိုင်းဖြစ်၏  -->
7                  <tr>
8                  <td><xsl:value-of select="booklist/book/name" /></td>
9                  <td><xsl:value-of select="booklist/book/price" /></td>
10                 </tr>
11         </table>
12         </body></html>
13 </xsl:template>
```

book.xml ၏ Book Node များအား အထက်ပါစည်းမျဉ်းအတိုင်း ပုံစံပြောင်းပေးမည် ဖြစ်သည်။ book.xml၏ စာအုပ်၃အုပ်လုံးသည် အထက်ပါစည်းမျဉ်းအတိုင်းပုံစံပြောင်း နေသည်ကို ကြည့်ရှုနိုင်ပါသည်။

3. Text Node များကိုအသုံးပြုခြင်း

book.xsl၏ စာကြောင်း၁၄နှင့် ၁၅ကိုကြည့်ပါ။ value-of Tag ကိုအသုံးပြုပြီး XML စာပိုက်၏ Text Node များကို ပုံစံပြောင်းပေးနေပါသည်။ XML စာပိုဒ်အတွင်းမှ select attribute တွင်သတ်မှတ်ထားသော Text Node အမည်ရှိသော Node များ၏တန်ဖိုးများအား ဤ Tag အတွင်းတွင် သက်မှတ်ထားသော စည်းမျဉ်းအတိုင်း ပုံစံပြောင်းပေးရမည်ဖြစ်၏။

```
<xsl:value-of select="name">
```

4. xsl: Prefix မဟုတ်သော Tag များ

   Template စည်းမျဉ်းအတွင်း xslနှင့်အစမပြုသော အစိတ်အပိုင်းများသည် ပုံစံပြောင်းရန်မလိုအပ်သော အစိတ်အပိုင်းများဖြစ်ပြီး၊ ရေးသားထားသည့်အတိုင်း ပုံစံပြောင်းပြီးသောစာပိုဒ်အတွင်းတွင် ပါဝင်ဖော်ပြသွားမည်ဖြစ်သည်။

3.4. XPath Functions

XPath တွင် xml အချက်အလက်များကို လိုအပ်သလို တွက်ချက်အသုံးပြုနိုင်ရန် Function များစွာ ရှိကြပါသည်။ အောက်ပါနမူနာဖြင့် XPath Function များ၏အသုံးပြုပုံကို ဖော်ပြသွားပါမည်။

book.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <?xml-stylesheet href="book.xsl" type="text/xsl"?>
3. <booklist>
4.       <book>
5.             <name>Head First Java</name>
6.             <price>25.00</price>
7.       </book>
8.       <book>
9.             <name>Effective Java</name>
10.                  <price>42.09</price>
11.            </book>
12.            <book>
13.                  <name>Head First Design Patterns</name>
14.                  <price>29.12</price>
15.            </book>
16.            <book>
17.                  <name>SQL Hacks</name>
18.                  <price>18.36</price>
19.            </book>
20.            <book>
21.                  <name>EJB 3.1</name>
22.                  <price>31.00</price>
23.            </book>
24.            <book>
25.                  <name>Seasor 2 Notes</name>
26.                  <price>19.28</price>
27.            </book>
28.            <book>
29.                  <name>Java Remote Interface</name>
30.                  <price>29.03</price>
31.            </book>
32.      </booklist>
```

- Count Function

  book.xsl၏ စာကြောင်းနံပါတ်၉ကိုကြည့်ပါ။ select attribute တွင် count(booklist/book) ဟုရေးသား ထားပါသည်။ ထိုစာကြောင်းတွင် count function ကိုအသုံးပြုပြီး၊ booklist ထဲမှစာအုပ်များကို ရေတွက် နေပါသည်။

- Sum Function

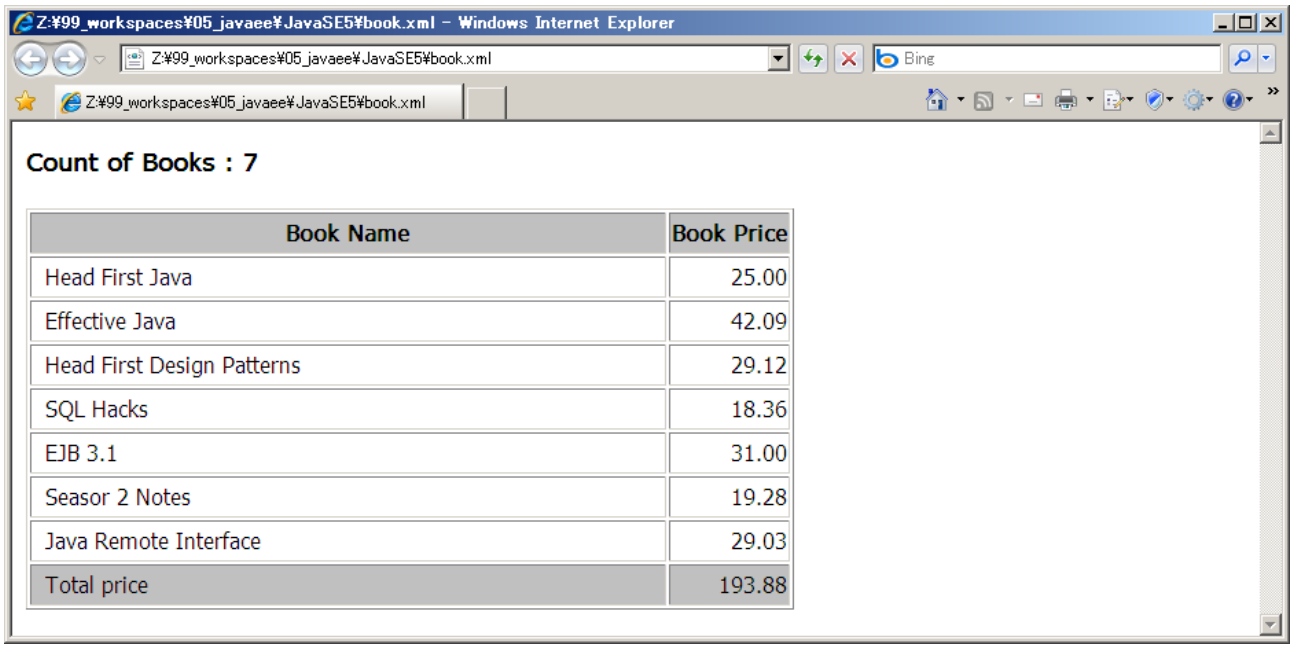
  book.xsl၏ စာကြောင်းနံပါတ် ၂၀ကိုကြည့်ပါ။ select attribute တွင် sum(booklist/book/price) ဟုရေးသား ထားပါသည်။ sum function ကိုသုံးပြီး စာအုပ်ဈေးနှုန်း စုပေါင်းဈေးနှုန်းကို တွက်ယူ နေပါသည်။

book.xsl

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

```
2. <xsl:stylesheet version="1.0"
3.       xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4.       <xsl:template match="/">
5.             <h3>
6.             Count of Books :
7.             <xsl:value-of select="count(booklist/book)" /></h3>
8.             <table border="1">
9.             <tr style="background-color:silver">
10.                  <th>Book Name</th>
11.                  <th>Book Price</th>
12.                  </tr>
13.                  <xsl:apply-templates select="booklist/book" />
14.                  <tr style="background-color:silver">
15.                  <td style="padding-left:10px"> Total price</td>
16.                  <td style="text-align: right">
17.                  <xsl:value-of select="sum(booklist/book/price)" />
18.                  </td>
19.                  </tr>
20.                  </table>
21.            </xsl:template>
22.            <xsl:template match="book">
23.                  <tr><td style="padding-left:10px;width:450px">
24.                        <xsl:value-of select="name"></xsl:value-of></td>
25.                  <td style="text-align: right">
26.                        <xsl:value-of select="price"></xsl:value-of>
27.                  </td></tr>
28.            </xsl:template>
29.      </xsl:stylesheet>
```

book.xml ကို web browserဖြင့် ဖွင့်ကြည့်သောအခါ အောက်ပါအတိုင်းတွေ့မြင်ရမည်ဖြစ်သည်။



Book Name	Book Price
Head First Java	25.00
Effective Java	42.09
Head First Design Patterns	29.12
SQL Hacks	18.36
EJB 3.1	31.00
Seasor 2 Notes	19.28
Java Remote Interface	29.03
Total price	193.88

အထက်ပါ countနှင့်sum အပြင် XPath တွင် အသုံးပြုရန် function များစွာရှိပါသည်။ အသုံးပြုသည့် အချက်အလက် များပေါ်တွင်မူတည်ပြီး function များကို အဖွဲ့အစည်းခွဲခြားထားပြီး အသုံးများသော functions

များမှာ အောက်ပါအတိုင်းဖြစ်၏။

အမည်	အသုံးပြုပုံ
Node Set Functions	
count (node set)	Node set၏ အရေအတွက်ကို သိလိုသည့်အခါမှာ အသုံးပြုရပါသည်။
id (identifier name)	id ၏တန်ဖိုးသည် identifier name နှင့်တူညီသော Element များကိုရှာဖွေနိုင်ပါသည်။
last ()	လက်ရှိ Node၏ နောက်ဆုံးနေရာတွင်ရှည်သော Element ကို ရရှိနိုင်ပါသည်။
name (node set)	Node Set၏ အမည်ကိုရရှိနိုင်ပါသည်။
position ()	လက်ရှိ Node၏ တည်နေရာကို ရရှိနိုင်ပါသည်။
String Functions	
concat (String1, String2, ...)	စကားလုံးစုများကို ပေါင်းစပ်ရာတွင် အသုံးပြုနိုင်ပါသည်။
substring (String, index, count)	စကားလုံးစု String မှ၊ indexနေရာမှစ၍ count စာလုံးမြောက်အထိ ဖြတ်ယူနိုင်သည်။
substring-after(String1, String2)	String1 အတွင်းရှိ String2 စာလုံး၏ နောက်မှကျန်သော စကားလုံးကိုရရှိနိုင်ပါသည်။
substring-before(String1, String2)	String1 အတွင်းရှိ String2 စာလုံး၏ ရှေ့တွင်ရှိသော စကားလုံးကိုရရှိနိုင်ပါသည်။
substring-length(String)	စကားလုံးစု၏ စာလုံးအရေအတွက်ကို ရရှိနိုင်ပါသည်။
translate(String1, String2, String3)	String1 မှ String2 များကို String3 အဖြစ်အစားထိုး ဖော်ပြပေးနိုင်ပါသည်။
Numeric Functions	
number (String)	String စကားလုံးစုများမှ ကိန်းဂဏန်းအဖြစ် ပြောင်းလည်းယူနိုင်ပါသည်။
round (numeric)	ကိန်းဂဏန်းကို အနည်းစပ်ဆုံး အပြည့်ကိန်းများအဖြစ် ပြောင်းလည်းပေးနိုင် ပါသည်။
sum (node set)	Node set များ၏ စုစုပေါင်း ရလဒ်ကို တွက်ယူနိုင်ပါသည်။

XPath Function များသည် XSL တွင်သာမက XQuery တွင်လည်း အသုံးပြုနိုင်ပါသည်။ အသေးစိတ်ကိုတော့ W3C ရဲ့ Web Page များတွင်လေ့လာနိုင်ပါသည်။

http://www.w3.org/TR/xpath-functions/

3.5. အသ XPath Operators

အထက်ပါအခန်းများတွင် ဖော်ပြပြီးသည့်အတိုင်း XSL တွင် XPath၏ဖော်ပြချက်များအပေါ် မူတည်၍ သက်ဆိုင်ရာ Node များ၏ Style များကို ပြောင်းလည်းပေးခြင်း၊ တဖန် XPath၏ Functions များကို အသုံးပြုပြီး သက်ဆိုင်ရာ Node များမှ လိုအပ်သော တန်ဖိုးများကို ပြောင်းလည်းယူခြင်းများကို ပြုလုပ်နိုင်ကြောင်း တင်ပြခဲ့ပါသည်။ တဖန် XPath၏ ဖော်ပြချက်များကို လိုအပ်သလို အသုံးပြုနိုင်စေရန် Operators များကိုလည်း ပြင်ဆင်ထားပါသည်။ အသုံးပြုနိုင်သော Operators များမှာ အောက်ပါအတိုင်းဖြစ်၏။

Operators	ရှင်းလင်းချက်နှင့်အသုံးပြုပုံများ
|	Node Set များကို၊ ပေါင်းဆက်ရာတွင် အသုံးပြုပါသည်။ **// book | //cd** ဟုရေးထားမည်ဆိုလျှင်၊ current node ၏အောက်တွင်ရှိသော book node များနှင့်၊ cd node များကို node set တစ်ခုအဖြစ်ရရှိမည် ဖြစ်သည်။
+	အပေါင်းလက္ခဏာ ဖြစ်ပါသည်။ count (//book) + count (//cd) ဟုရေးသားထားမည်ဆိုလျှင် စာအုပ်အရေအတွက်နှင့် CD အရေအတွက်ကို စုစုပေါင်းအရေအတွက်ကို ရရှိမည်ဖြစ်သည်။
-	အနှုတ်လက္ခဏာ ဖြစ်ပါသည်။
*	အမြှောက်လက္ခဏာ ဖြစ်ပါသည်။
div	အစားလက္ခဏာ ဖြစ်ပါသည်။
mod	အကြွင်းကိုသိလိုသောအခါ အသုံးပြုနိုင်ပါသည်။
=	တူညီခြင်းကို ဖော်ပြလိုသောအခါ အသုံးပြုနိုင်ပါသည်။ //book/price=39.00 ဟုရေးသားထားလျှင် book၏တန်ဖိုးသည် 39.00 ရှိသော စာအုပ်များဟု အဓိပ္ပါယ်ရပါသည်။
!=	မတူညီခြင်းကို ဖော်ပြလိုသောအခါ အသုံးပြုနိုင်ပါသည်။ //book/price=39.00 ဟုရေးသားထားလျှင် book၏တန်ဖိုးသည် 39.00 မဟုတ်သော စာအုပ်များဟု အဓိပ္ပါယ်ရပါသည်။
<	ပို၍သေးငယ်ခြင်း ကိုဖော်ပြပါသည်။
<=	ပို၍သေးငယ်ခြင်း သို့မဟုတ် တူညီခြင်း ကိုဖော်ပြပါသည်။
>	ပို၍ကြီးမားခြင်း ကိုဖော်ပြပါသည်။
>=	ပို၍ကြီးမားခြင်း သို့မဟုတ် တူညီခြင်း ကိုဖော်ပြပါသည်။
or	သို့မဟုတ်။ A or B ဟုရေးသားထားလျှင် A သို့မဟုတ် B ဟုအဓိပ္ပါယ်ရပါသည်။
and	နှင့်။ A or B ဟုရေးသားထားလျှင် A နှင့် B ဟုအဓိပ္ပါယ်ရပါသည်။

3.6. အခြားသော ဝါကျများ

3.6.1 IF

အခြားသော Programming ဘာသာရပ်များကဲ့သို့ပင် if ကို အသုံးပြုနိုင်ပါသည်။ ဖော်ပြပါအနေအထား

အတိုင်းပြည့်ဆုံမည် ဆိုလျှင် ရေးသားထားသော လုပ်ဆောင်ချက်များကို လုပ်ဆောင်သွားမည်ဖြစ်သည်။ သို့ရာတွင် သတိထားရမည်မှာ else ကဲ့သို့ အသုံးပြုချက်များမှာ XPath တွင်အသုံးပြု၍မရပေ။

world.xml

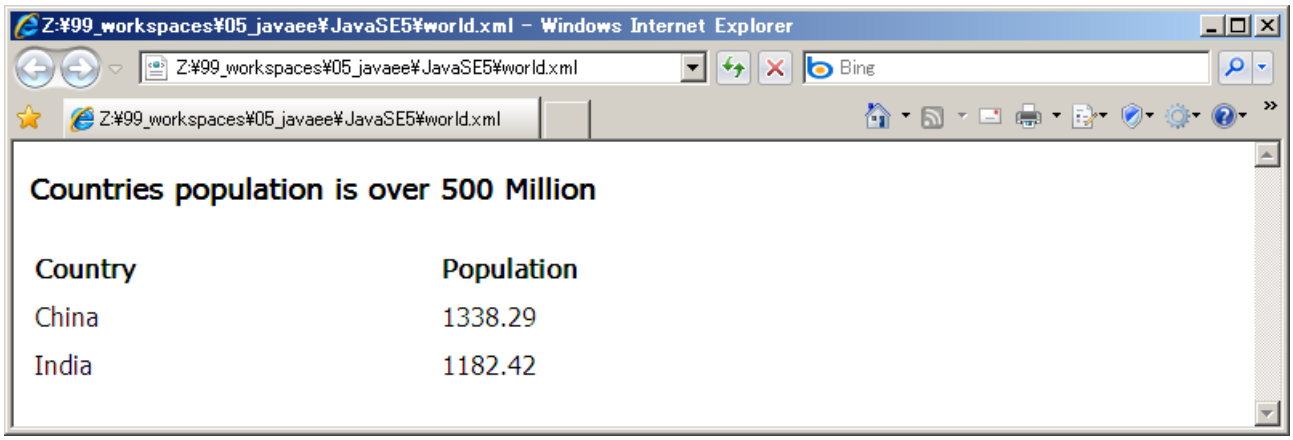
```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <?xml-stylesheet href="world.xsl" type="text/xsl" ?>
3. <world>
4. <country population="1338.29">China</country>
5. <country population="1182.42">India</country>
6. <country population="309.59">USA</country>
7. <country population="127.53">Japan</country>
8. <country population="42.642">Myanmar</country>
9. </world>
```

world.xsl

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xsl:stylesheet version="1.0"
3.       xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4.
5.       <xsl:template match="/">
6.             <h3>Countries population is over 500 Million</h3>
7.             <table>
8.                   <tr>
9.                         <th style="text-align:left">Country</th>
10.                        <th>Population</th>
11.                  </tr>
12.                  <xsl:apply-templates select="world/country" />
13.            </table>
14.      </xsl:template>
15.
16.      <xsl:template match="country">
17.            <xsl:if test="./@population > 500">
18.                  <tr>
19.                        <td width="250px">
20.                              <xsl:value-of select="." />
21.                        </td>
22.                        <td>
23.                              <xsl:value-of select="./@population" />
24.                        </td>
25.                  </tr>
26.            </xsl:if>
27.      </xsl:template>
28.
29.      </xsl:stylesheet>
```

World.xsl၏ စာကြောင်းဝဂ်မှ ၂၆အထိသည် If ဝါကျကို အသုံးပြုထားသော နမူနာဖြစ်သည်။ စာကြောင်းဝဂ်တွင် **<xsl:if test="./@population > 500">** ဟူ၍ If ဝါကျကိုရေးသားထား၏။ test attribute သည် If ဝါကျတွင် မရေးမဖြစ် ပါဝင်ရမည်ဖြစ်ပြီး အသုံးပြုမည့် ဖော်ပြချက်လည်းဖြစ်၏။ **./@population > 500** သည် လက်ရှိ node၏ population attribute ရဲ့ တန်ဖိုးသည် 500 အထက်ဖြစ်လျှင်ဟု အဓိပ္ပါယ်ရပါသည်။ အကယ်၍ test အတွင်းကဖော်ပြချက်နှင့် ကိုက်ညီမည်ဆိုလျှင် IF ဝါကျ အတွင်းမှ ရေးသားထားသော သတ်မှတ်ချက်များအတိုင်း ပြောင်းလည်းပေးသွားမည်ဖြစ်သည်။ အထက်ပါ world.xml ကို web browserနှင့်

ဖွင့်ကြည့်မည် ဆိုလျှင် အောက်ပါအတိုင်းတွေ့မြင်ရမည် ဖြစ်သည်။



3.6.2 Choose

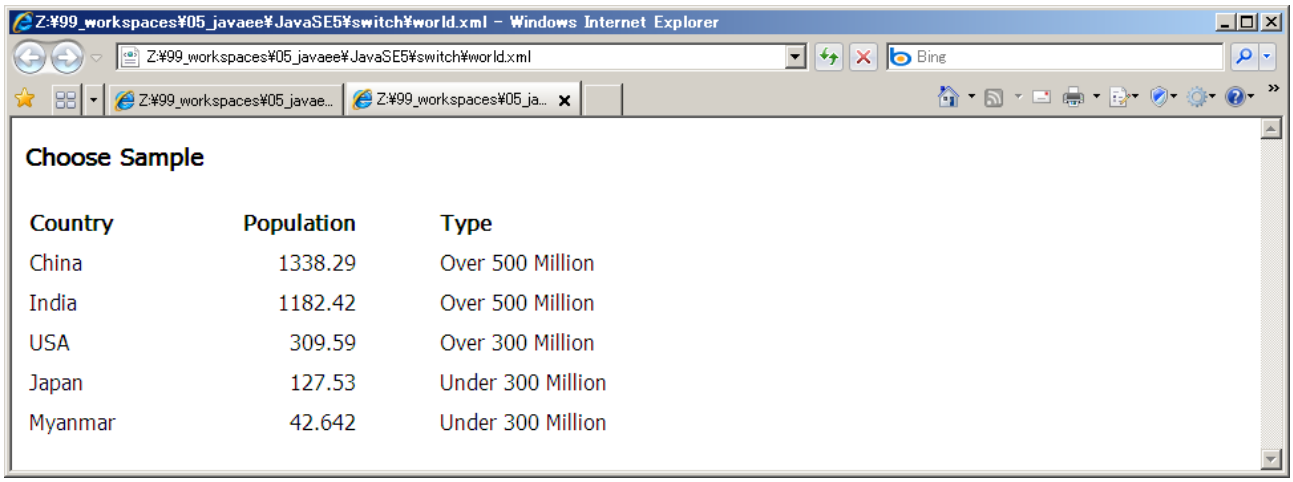
နှိုင်းယှဉ်ရန် အချက်အလက်သည် တစ်ခုထက်ပိုရှိပြီး၊ အချက်အလက်များကို ကိုယ်စီလုပ်ဆောင်ချက်များကို ခွဲခြားပြီး ရေးသားလိုလျှင် အသုံးပြုနိုင်ပေသည်။ Choose node ၏ Child nodeဖြစ်သော When Node များတွင် အချက်အလက်များကို သီးခြားရေးသားနိုင်ပေသည်။ အခြားသော Programming ဘာသာရပ်များ၏ switch ဝါကျများနှင့် အသုံးပြုပုံတူညီပေသည်။

world.xsl

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xsl:stylesheet version="1.0"
3.       xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4.
5.       <xsl:template match="/">
6.             <h3>Choose Sample</h3>
7.             <table width="60%">
8.                   <tr>
9.                         <th style=
10.                                     "text-align:left">Country</th>
11.                               <th style=
12.                                     "text-align:right">Population</th>
13.                               <th style=
14.                                     "text-align:left;
15.                                     padding-left:60px">Type</th>
16.                         </tr>
17.                         <xsl:apply-templates select="world/country" />
18.                   </table>
19.             </xsl:template>
20.
21.             <xsl:template match="country">
22.                   <tr>
23.                         <td>
24.                               <xsl:value-of select="." />
25.                         </td>
26.                         <td style="text-align:right">
27.                               <xsl:value-of select="@population" />
```

```
28.                         </td>
29.                         <td style="text-align:left;padding-left:60px">
30.                               <xsl:choose>
31.                                     <xsl:when test="@population > 500">
32.                                           Over 500 Million
33.                                     </xsl:when>
34.                                     <xsl:when test="@population > 300">
35.                                           Over 300 Million
36.                                     </xsl:when>
37.                                     <xsl:otherwise>
38.                                           Under 300 Million
39.                                     </xsl:otherwise>
40.                               </xsl:choose>
41.                         </td>
42.                   </tr>
43.             </xsl:template>
44.
45. </xsl:stylesheet>
```

If နမူနာတွင် ရေးသားထားသော world.xsl အား အထက်ပါအတိုင်းပြောင်းလည်း ရေးသားကြည့်ပါမည်။ စာကြောင်းဂဏန်း ၄ဝမှ ၄ဝအထိသည် choose ဝါကျကို အသုံးပြုထားပါသည်။ When node အတွင်းမှ test သတ်မှတ်ချက်များနှင့် ပြည့်စုံလျှင် When အတွင်းတွင်ရေးသားထားသော လုပ်ဆောင်ချက်များအတိုင်း လုပ်ဆောင်သွားမည် ဖြစ်သည်။ တဖန် When လုပ်ဆောင်ချက်များနှင့် မပြည့်စုံပါက Otherwise node အတွင်းမှ လုပ်ဆောင်ချက်ကို လုပ်ဆောင်ပေးမည် ဖြစ်၏။



3.6.3 For each

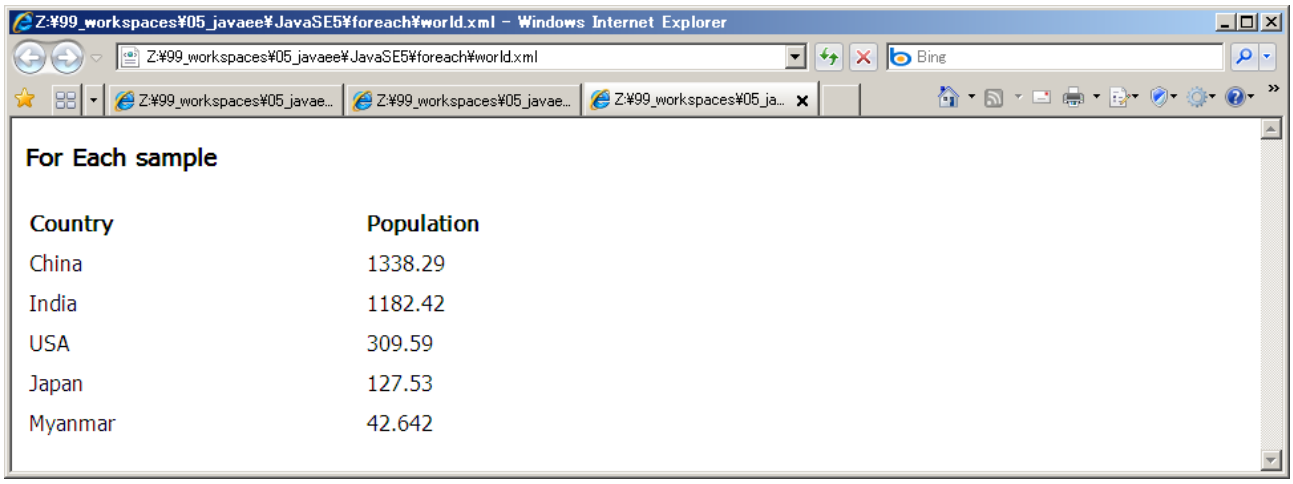
Node List များမှ Node များကိုတစ်ခုပြီးတစ်ခု သတ်မှတ်ထားသည့် လုပ်ဆောင်ချက်များကို လုပ်ဆောင်စေလိုလျှင် for-each ကို အသုံးပြုရမည်ဖြစ်သည်။ အခြားသော Programming ဘာသာရပ်များမှ while သို့မဟုတ် for ဝါကျနှင့် ဆင်တူပါသည်။

world.xsl

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xsl:stylesheet version="1.0"
3.       xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4.
```

```
5.      <xsl:template match="/">
6.            <h3>For Each sample</h3>
7.            <table>
8.                  <tr>
9.                        <th style="text-align:left">Country</th>
10.                             <th>Population</th>
11.                       </tr>
12.                       <xsl:for-each select="world/country">
13.                       <tr>
14.                             <td>
15.                                   <xsl:value-of select="." />
16.                             </td>
17.                             <td>
18.                                   <xsl:value-of select="@population" />
19.                             </td>
20.                       </tr>
21.                       </xsl:for-each>
22.                 </table>
23.           </xsl:template>
24.
25. </xsl:stylesheet>
```

wrold.xsl file ကို အထက်ပါအတိုင်းပြောင်းလည်း ရေးသားပြီး Web Browserဖြင့် ဖွင့်ကြည့်မည် ဆိုလျှင် အောက်ပါအတိုင်းတွေ့မြင်ရမည် ဖြစ်သည်။



3.6.4 Sort

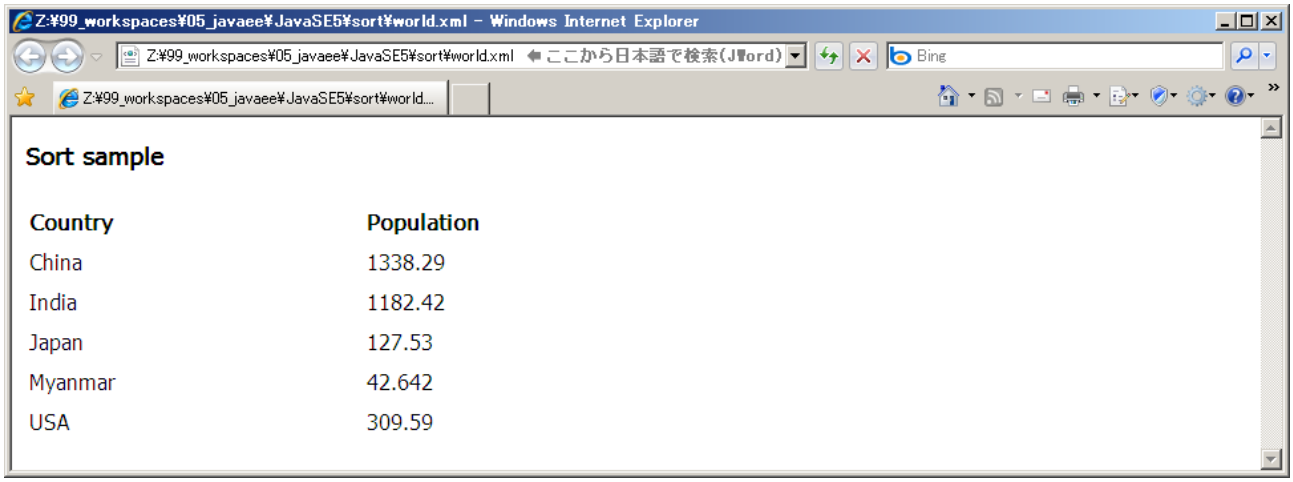
ဖော်ပြပုံအစီအစဉ်များကို အစီအစဉ်တကျ ပြောင်းလည်း စီစဉ်လိုပါက Sort ဝါကျကို အသုံးပြုရမည် ဖြစ်သည်။ Sort Element သည် <xsl:apply-templates> သို့မဟုတ် <xsl:for-each> Element များနှင့် တွဲဖက်၍ အသုံးပြုလေ့ရှိပါသည်။ Sort element တွင် သတ်မှတ်နိုင်သော Attribute များမှာ အောက်ပါ အတိုင်းဖြစ်၏။

အမည်	ရှင်းလင်းချက်
select	အစီအစဉ်ပြောင်းလည်းရာတွင် အသုံးပြုမည့် Node Set ကို ရေးသားရမည်။
lang	အစီအစဉ်ပြောင်းလည်းရာတွင် အသုံးပြုမည့် ဘာသာစကားကို ရေးသားရမည်။

data-type	အစီအစဉ်ပြောင်းလည်းရာတွင် အသုံးပြုမည့် data အမျိုးအစားကို ရေးသားရမည် ဖြစ်သည်။ • text : စာလုံးအမျိုးအစာ data • number : ကိန်းဂဏန်းအမျိုးအစား data
order	အစီအစဉ်ပြောင်းလည်းရာတွင် အသုံးပြုမည့် အစီအစဉ်ကို ရေးသားရမည် ဖြစ်သည်။ • ascending : အတက်အစီအစဉ် • descending : အကျအစီအစဉ်
case-order	data-tpye က Textဖြစ်မှသာလျှင် အသုံးပြုရမည် ဖြစ်သည်။ • upper-first : စာလုံးအကြီးများကို အရင်ဖော်ပြလိုသည့်အခါ • lower-first : စာလုံးအငယ်များကို အရင်ဖော်ပြလိုသည့်အခါ

For-each နမူနာတွင်အသုံးပြုခဲ့သော world.xsl file ၏ စာကြောင်း ၁၂မှ ၁၃အကြားတွင် အောက်ပါ စာကြောင်းကို ဖြည့်စွက်လိုက်မည် ဆိုပါက အစီအစဉ်များကို အမည်အတိုင်း ပြောင်းလည်း စီစဉ်ပေးသွားနိုင်မည် ဖြစ်ပေသည်။

```
<xsl:sort select="." data-type="text" order="ascending"/>
```



3.6.5 Attributes များကိုသတ်မှတ်ခြင်း

xsl တွင် ပြောင်းလည်းရန် Element များကပိုင်ဆိုင်သော Attributes များ၏ တန်ဖိုးများကို သတ်မှတ်ပေးနိုင်ပါသည်။ ထိုအခါတွင် အသုံးပြုသော ဝါကျမှာ attributeဖြစ်၏။

attr.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <?xml-stylesheet href="attr.xsl" type="text/xsl" ?>
3. <pages>
4. <page url="http://caloricp.jp/apc/sscp/K1006CAL/top.html">Caloricp</page>
5. <page
   usl="http://www.ajinomoto.co.jp/aji/egg/generator2/">Ajinomoto</page>
6. <page url="http://manifani.com/">Manifani</page>
7. <page url="http://www.brucke.co.jp/kolok/">Brucke</page>
```

```
8. <page url="http://aikohno.com/">Aikohno</page>
9. <page url="http://www.premacy.mazda.co.jp/pre/">Masada Premacy</page>
10.      </pages>
```

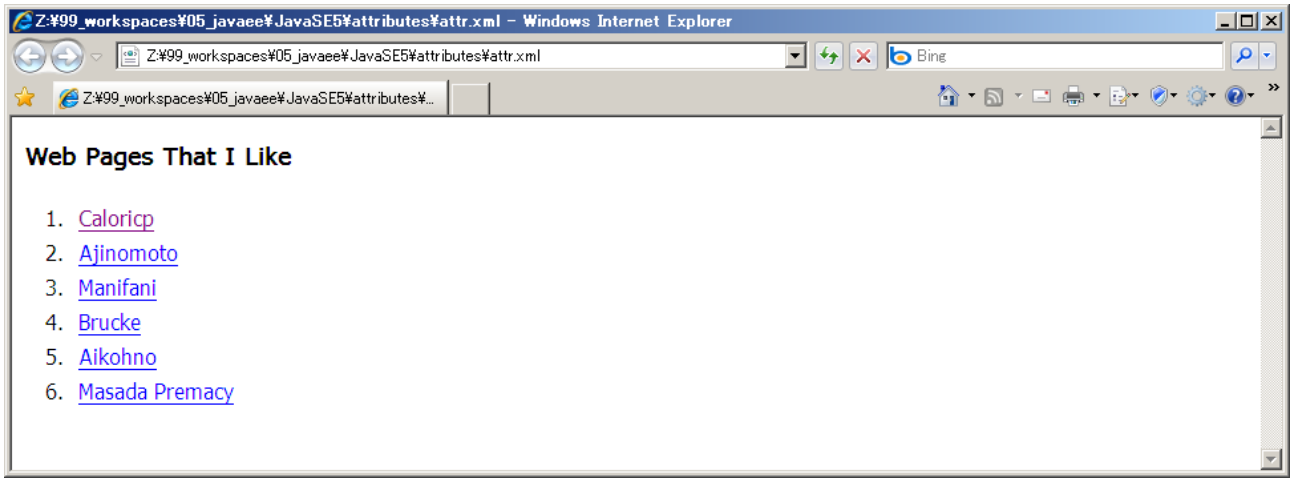
attr.xsl

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xsl:stylesheet version="1.0"
3. xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4.       <xsl:template match="/">
5.             <h3>Web Pages That I Like</h3>
6.             <ol>
7.             <xsl:for-each select="pages/page">
8.                   <li>
9.                         <a>
10.                              <xsl:attribute name="href">
11.                              <xsl:value-of select="@url" />
12.                              </xsl:attribute>
13.                              <xsl:value-of select="." />
14.                         </a>
15.                  </li>
16.            </xsl:for-each>
17.            </ol>
18.      </xsl:template>
19. </xsl:stylesheet>
```

အထက်ပါ attr.xslဏ် စာကြောင်းနံပါတ်၁၀မှ ၁၂အထိသည် Attribute ကိုသတ်မှတ်ပေးသော ဝါကျဖြစ်ဏ်။ စာကြောင်း၉တွင်ရေးသားထားသော Element <A> ဏ် href ဟု အမည်ရသော Attrubute ဏ် တန်ဖိုးကို သတ်မှတ်ပေးနေပါသည်။ တန်ဖိုးမှာ စာကြောင်းနံပါတ် ၁၁ တွင်ရေးသားထားသည့်အတိုင်း လက်ရှိ Nodeဏ် url Attributeဏ် တန်ဖိုးဖြစ်ဏ်။ ထိုသို့ရေးသားထားခြင်းအားဖြင့် <A> element ဏ် href attribute တန်ဖိုးတွင် page node ဏ် url attribute ဏ် တန်ဖိုးကို အစားထိုးဖော်ပြပေးနိုင်ပါသည်။

attr.xml file ကို web browserဖြင့် ဖွင့်ကြည့်သောအခါ တဖက်စာမျက်နှာအတိုင်း တွေ့မြင်ရမည်ဖြစ်ဏ်။

attr.xml



3.7. ပြန်လည်သုံးသပ်ချက်

ဤ XML ဆီသို့ခြေလှမ်းများရဲ့ တတိယခြေလှမ်းဖြစ်တဲ့ အခြေခံXSL တွင် XSLT ကိုအသုံးပြုပြီး XML စာပိုဒ်များ ၏ တည်ဆောက်ပုံကို အခြားသောတည်ဆောက်ပုံအဖြစ် ပြောင်းလည်းနိုင်ပုံကို ဖော်ပြခဲ့ပါသည်။ ဤအခန်းတွင် HTML သို့ပြောင်းလည်းပုံကို အဓိကထားဖော်ပြခဲ့ပေမယ့်၊ XML စာပိုဒ်အတူတူခြင်းဖြစ်လင့်ကစား Style Sheet ကို ပြောင်းလည်းခြင်းအားဖြင့် ပုံစံတစ်မျိုးဆီသို့ပြောင်းလည်းပေးနိုင်ကြောင်း နားလည်နိုင်ပေလိမ့်မည်။ ယခုအခန်းတွင် XSLT Processor အဖြစ် Internet Explore ကို အသုံးပြုခဲ့ပါသဖြင့် ပြောင်းလည်းပြီးနောက် အချက်အလက်များ ကိုပြန်လည် အသုံးပြုခြင်းမရှိခဲ့ပေမယ့်၊ Jax သို့မဟုတ် Xalan ကဲ့သို့ XSLT Processor များကို အသုံးပြုပါက ပြောင်းလည်းပြီးသော အချက်အလက်များ File အဖြစ် ထုတ်ပြန်ပေးနိုင်ပါ၏။ တဖန် Style Sheet၏ ရေးသား ပုံပေါ်မှုတည်ပြီး CSV ပုံသဏ္ဌာန် အဖြစ်ရေးသားနိုင်ပါသဖြင့် XML အချက်အလက်များကို Microsoft Excel များမှ တဆင့် အသုံးပြုနိုင်ပေးမည်ဖြစ်သည်။

နောက်အခန်းများတွင် DOM, SAX API ကိုအသုံးပြု၍ XML အချက်အလက်များကို Java Program မှတဆင့် အသုံးပြုပုံများကို ဆက်လက်ဖော်ပြသွားမည် ဖြစ်သည်။