基本的XPath语法类似于在一个文件系统中定位文件,如果路径以斜线 / 开始, 那么该路径就表示到一个元素的绝对路径

/AAA

选择根元素AAA

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/CCC

选择AAA 的所有CCC子元素

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/DDD/BBB

选择AAA 的子元素DDD的所有子元素

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

如果路径以双斜线 // 开头, 则表示选择文档中所有满足双斜线//之后规则的元素(无论层级关系)

//BBB

选择所有BBB元素

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC>
      <DDD>
         <BBB/>
         <BBB/>
      </DDD>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//DDD/BBB

选择所有父元素是DDD的BBB元素

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC>
      <DDD>
         <BBB/>
         <BBB/>
      </DDD>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

星号 * 表示选择所有由星号之前的路径所定位的元素

/AAA/CCC/DDD/*

选择所有路径依附于/AAA/CCC/DDD 的元素

```
<AAA>
   <XXX>
      <DDD>
         <BBB/>
         <BBB/>
         <EEE/>
         <FFF/>
      </DDD>
   </XXX>
   <CCC>
      <DDD>
         <BBB/>
         <BBB/>
         <EEE/>
         <FFF/>
      </DDD>
   </CCC>
   <CCC>
      <BBB>
         <BBB>
            <BBB/>
         </BBB>
      </BBB>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/*/*/*/BBB

选择所有的有3个祖先元素的BBB元素

```
<AAA>
   <XXX>
      <DDD>
         <BBB/>
         <BBB/>
         <EEE/>
         <FFF/>
      </DDD>
   </XXX>
   <CCC>
      <DDD>
         <BBB/>
         <BBB/>
         <EEE/>
         <FFF/>
```

```
        </ DDD>
      </ CCC>
      < CCC>
        < BBB>
          < BBB>
            < BBB/>
          </ BBB>
        </ BBB>
      </ CCC>
    </ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

//*

选择所有元素

```
<AAA >
  < XXX>
    < DDD>
      < BBB/>
      < BBB/>
      < EEE/>
      < FFF/>
    </ DDD>
  </ XXX>
  < CCC>
    < DDD>
      < BBB/>
      < BBB/>
      < EEE/>
      < FFF/>
    </ DDD>
  </ CCC>
  < CCC>
    < BBB>
      < BBB>
        < BBB/>
      </ BBB>
    </ BBB>
  </ CCC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

方块号里的表达式可以进一步的指定元素, 其中数字表示元素在选择集里的位置, 而last()函数则表示选择集中的最后一个元素.

/AAA/BBB[1]

选择AAA 的第一个BBB子元素

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/BBB[last()]

选择AAA 的最后一个BBB子元素

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

属性通过前缀 @ 来指定

//@id

选择所有的id属性

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[@id]

选择有id属性的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[@name]

选择有name属性的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[@*]

选择有任意属性的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[not(@*)]

选择没有属性的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```

在**XLab**中打开实例 | 树视图 **(JPG)**

属性的值可以被用来作为选择的准则, normalize-space函数删除了前部和尾部的空格, 并且把连续的空格串替换为一个单一的空格

//BBB[@id='b1']

选择含有属性id且其值为'b1'的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB name = " bbb "/>
   <BBB name = "bbb"/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[@name='bbb']

选择含有属性name且其值为'bbb'的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB name = " bbb "/>
   <BBB name = "bbb"/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[normalize-space(@name)='bbb']

选择含有属性name且其值(在用normalize-space函数去掉前后空格后)为'bbb'的BBB元素

```
<AAA>
   <BBB id = "b1"/>
   <BBB name = " bbb "/>
   <BBB name = "bbb"/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

count()函数可以计数所选元素的个数

//*[count(BBB)=2]

选择含有2个BBB子元素的元素

```
<AAA>
   <CCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </CCC>
   <DDD>
      <BBB/>
      <BBB/>
   </DDD>
   <EEE>
      <CCC/>
      <DDD/>
   </EEE>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[count(*)=2]

选择含有2个子元素的元素

```
<AAA>
   <CCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </CCC>
   <DDD>
      <BBB/>
      <BBB/>
   </DDD>
   <EEE>
      <CCC/>
      <DDD/>
   </EEE>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[count(*)=3]

选择含有3个子元素的元素

```
<AAA>
   <CCC>
      <BBB/>
```

```
        < BBB/>
        < BBB/>
    </ CCC>
    < DDD>
        < BBB/>
        < BBB/>
    </ DDD>
    < EEE>
        < CCC/>
        < DDD/>
    </ EEE>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

name()函数返回元素的名称, start-with()函数在该函数的第一个参数字符串是以第二个参数字符开始的情况返回true, contains()函数当其第一个字符串参数包含有第二个字符串参数时返回true.

//*[name()='BBB']

选择所有名称为BBB的元素(这里等价于//BBB)

```
<AAA>
   <BCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </BCC>
   <DDB>
      <BBB/>
      <BBB/>
   </DDB>
   <BEC>
      <CCC/>
      <DBD/>
   </BEC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[starts-with(name(),'B')]

选择所有名称以"B"起始的元素

```
<AAA>
   <BCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </BCC>
   <DDB>
      <BBB/>
      <BBB/>
   </DDB>
   <BEC>
      <CCC/>
      <DBD/>
   </BEC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[contains(name(),'C')]

选择所有名称包含"C"的元素

```
<AAA>
   <BCC>
```

```
        < BBB/>
        < BBB/>
        < BBB/>
    </ BCC>
    < DDB>
        < BBB/>
        < BBB/>
    </ DDB>
    < BEC>
        < CCC/>
        < DBD/>
    </ BEC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

string-length函数返回字符串的字符数,你应该用<替代<, 用>代替>

//*[string-length(name()) = 3]

选择名字长度为3的元素

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[string-length(name()) < 3]

选择名字长度小于3的元素

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//*[string-length(name()) > 3]

选择名字长度大于3的元素

```
<AAA>
   <Q/>
   <SSSS/>
   <BB/>
   <CCC/>
   <DDDDDDDD/>
   <EEEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

多个路径可以用分隔符 | 合并在一起

//CCC | //BBB

选择所有的CCC和BBB元素

```
<AAA>
   <BBB/>
   <CCC/>
   <DDD>
      <CCC/>
   </DDD>
   <EEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/EEE | //BBB

选择所有的BBB元素和所有是AAA 的子元素的EEE元素

```
<AAA>
   <BBB/>
   <CCC/>
   <DDD>
      <CCC/>
   </DDD>
   <EEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/EEE | //DDD/CCC | /AAA | //BBB

可以合并的路径数目没有限制

```
<AAA>
   <BBB/>
   <CCC/>
   <DDD>
      <CCC/>
   </DDD>
   <EEE/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

child轴(axis)包含上下文节点的子元素, 作为默认的轴,可以忽略不写.

/AAA

等价于 /child::AAA

```
<AAA>
   <BBB/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/child::AAA

等价于/AAA

```
<AAA>
   <BBB/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/BBB

等价于/child::AAA/child::BBB

```
<AAA>
   <BBB/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/child::AAA/child::BBB

等价于/AAA/BBB

```
<AAA>
   <BBB/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/child::AAA/BBB

二者都可以被合并

```
<AAA>
   <BBB/>
   <CCC/>
```

```
</AAA>
```

在**XLab**中打开实例 | 树视图 **(JPG)**

---

descendant (后代)轴包含上下文节点的后代,一个后代是指子节点或者子节点的子节点等等, 因此descendant轴不会包含属性和命名空间节点.

/descendant::*

选择文档根元素的所有后代.即所有的元素被选择

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

/AAA/BBB/descendant::*

选择/AAA/BBB 的所有后代元素

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在 中打开实例 树视图

在XLab中打开实例 | 树视图 (JPG)

//CCC/descendant::*

选择在祖先元素中有CCC的所有元素

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//CCC/descendant::DDD

选择所有以CCC为祖先元素的DDD元素

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

parent轴(axis)包含上下文节点的父节点, 如果有父节点的话

//DDD/parent::*

选择DDD元素的所有父节点

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

ancestor轴(axis)包含上下节点的祖先节点, 该祖先节点由其上下文节点的父节点以及父节点的父节点等等诸如此类的节点构成,所以ancestor轴总是包含有根节点,除非上下文节点就是根节点本身.

/AAA/BBB/DDD/CCC/EEE/ancestor::*

选择一个绝对路径上的所有节点

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//FFF/ancestor::*

选择FFF元素的祖先节点

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

following-sibling轴(axis)包含上下文节点之后的所有兄弟节点

/AAA/BBB/following-sibling::*

```
<AAA>
   <BBB>
      <CCC/>
      <DDD/>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//CCC/following-sibling::*

```
<AAA>
   <BBB>
      <CCC/>
      <DDD/>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

preceding-sibling 轴(axis)包含上下文节点之前的所有兄弟节点

/AAA/XXX/preceding -sibling::*

```
<AAA>
   <BBB>
      <CCC/>
      <DDD/>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//CCC/preceding-sibling::*

```
<AAA>
   <BBB>
      <CCC/>
      <DDD/>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

following 轴(axis)包含同一文档中按文档顺序位于上下文节点之后的所有节点, 除了祖先节点,属性节点和命名空间节点

/AAA/XXX/following::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
         <DDD>
            <EEE/>
         </DDD>
      </ZZZ>
      <FFF>
         <GGG/>
      </FFF>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//ZZZ/following::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
         <DDD>
            <EEE/>
         </DDD>
      </ZZZ>
      <FFF>
         <GGG/>
      </FFF>
   </BBB>
   <XXX>
```

```
        < DDD>
           < EEE/>
           < DDD/>
           < CCC/>
           < FFF/>
           < FFF>
              < GGG/>
           </ FFF>
        </ DDD>
     </ XXX>
     < CCC>
        < DDD/>
     </ CCC>
  </ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

following 轴(axis)包含同一文档中按文档顺序位于上下文节点之前的所有节点, 除了祖先节点,属性节点和命名空间节点

/AAA/XXX/preceding::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/preceding::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
```

```
    </ CCC>
  </ AAA >
```

在**XLab**中打开实例 | 树视图 **(JPG)**

descendant-or-self 轴(axis)包含上下文节点本身和该节点的后代节点

/AAA/XXX/descendant-or-self::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//CCC/descendant-or-self::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
```

```
</AAA>
```

在**XLab**中打开实例 | 树视图 **(JPG)**

---

ancestor-or-self 轴(axis)包含上下文节点本身和该节点的祖先节点

/AAA/XXX/DDD/EEE/ancestor-or-self::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/ancestor-or-self::*

```
<AAA>
   <BBB>
      <CCC/>
      <ZZZ>
         <DDD/>
      </ZZZ>
   </BBB>
   <XXX>
      <DDD>
         <EEE/>
         <DDD/>
         <CCC/>
         <FFF/>
         <FFF>
            <GGG/>
         </FFF>
      </DDD>
   </XXX>
   <CCC>
      <DDD/>
   </CCC>
```

```
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

---

ancestor, descendant, following, preceding 和self轴(axis)分割了XML文档(忽略属性节点和命名空间节点), 不能交迭, 而一起使用则包含所有节点

//GGG/ancestor::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/descendant::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
```

```
      </ DDD>
   </ XXX>
   < CCC>
      < DDD/>
   </ CCC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/following::*

```
<AAA >
   < BBB>
      < CCC/>
      < ZZZ/>
   </ BBB>
   < XXX>
      < DDD>
         < EEE/>
         < FFF>
            < HHH/>
            < GGG>
               < JJJ>
                  < QQQ/>
               </ JJJ>
               < JJJ/>
            </ GGG>
            < HHH/>
         </ FFF>
      </ DDD>
   </ XXX>
   < CCC>
      < DDD/>
   </ CCC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/preceding::*

```
<AAA >
   < BBB>
      < CCC/>
      < ZZZ/>
   </ BBB>
   < XXX>
      < DDD>
         < EEE/>
         < FFF>
            < HHH/>
            < GGG>
               < JJJ>
                  < QQQ/>
               </ JJJ>
               < JJJ/>
            </ GGG>
            < HHH/>
         </ FFF>
```

```
      </ DDD>
   </ XXX>
   < CCC>
      < DDD/>
   </ CCC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/self::*

```
<AAA >
   < BBB>
      < CCC/>
      < ZZZ/>
   </ BBB>
   < XXX>
      < DDD>
         < EEE/>
         < FFF>
            < HHH/>
            < GGG>
               < JJJ>
                  < QQQ/>
               </ JJJ>
               < JJJ/>
            </ GGG>
            < HHH/>
         </ FFF>
      </ DDD>
   </ XXX>
   < CCC>
      < DDD/>
   </ CCC>
</ AAA >
```

在XLab中打开实例 | 树视图 (JPG)

//GGG/ancestor::* | //GGG/descendant::* | //GGG/following::* | //GGG/preceding::* | //GGG/self::*

```
<AAA >
   < BBB>
      < CCC/>
      < ZZZ/>
   </ BBB>
   < XXX>
      < DDD>
         < EEE/>
         < FFF>
            < HHH/>
            < GGG>
               < JJJ>
                  < QQQ/>
               </ JJJ>
               < JJJ/>
            </ GGG>
            < HHH/>
```

```
            </ FFF>
         </ DDD>
      </ XXX>
      < CCC>
         < DDD/>
      </ CCC>
   </ AAA >
```

在**XLab**中打开实例 | 树视图 **(JPG)**

div运算符做浮点除法运算, mod运算符做求余运算, floor函数返回不大于参数的最大整数(趋近于正无穷), ceiling返回不小于参数的最小整数(趋近于负无穷)

//BBB[position() mod 2 = 0]

选择偶数位置的BBB元素

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <CCC/>
   <CCC/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//BBB[position() = floor(last() div 2 + 0.5) or position() = ceiling(last() div 2 + 0.5)]

选择中间的BBB元素

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <BBB/>
   <CCC/>
   <CCC/>
   <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)

//CCC[position() = floor(last() div 2 + 0.5) or position() = ceiling(last() div 2 + 0.5)]

选择中间的CCC元素

```
<AAA>
   <BBB/>
   <BBB/>
   <BBB/>
```

```
    <BBB/>
    <BBB/>
    <BBB/>
    <BBB/>
    <BBB/>
    <CCC/>
    <CCC/>
    <CCC/>
</AAA>
```

在XLab中打开实例 | 树视图 (JPG)