บทที่ 6

XML

ธีระยุทธ ทองเครือ

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น

ภาษา XML

- XML ย่อมาจาก Extensible Markup Language

- เป็นภาษาที่ใช้ในการกำหนดโครงสร้างของข้อมูลหรือเอกสาร

- มีโครงสร้างในแบบลำดับชั้นที่ชัดเจนแบบ Tree

- ข้อมูลที่มีอยู่จะถูกอธิบายความหมายด้วยการกำกับ (Markup) ด้วยแท็ก (Tag)
  - <firstName>John</firstname>
  - <age>15</age>

ส่วนประกอบของ XML

ส่วนหัว (Prolog)

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

ส่วนเนื้อหา

(Document Body)

```
<albums>
<cd>
   <title>Kind of Blue</title>
   <artist>Miles Davis</artist>
   <track length="9.22">So What</track>
   <track length="5.37">Blue in Green</track>
   <track length="11.33">All Blues</track>
</cd>
<cd>
   <title>Cookin</title>
   <artist>Miles Davis</artist>
   <track length="5.57">My Funny Valentine</track>
   <track length="9.53">Blues by Five</track>
</cd>
</albums>
```

Element

❖ Element คือ การกำหนดขอบเขตของข้อมูล

- Closed Element มีรูปแบบดังนี้

```
<element_name>ข้อมูล</element_name>
```

- Empty Element มีรูปแบบดังนี้

```
<element_name />
```

❖ ตัวอย่างของ Element เช่น

```
<artist>สิงโต นำโชค</artist>
```

Element

❖ Element เป็น case sensitive

```
<artist>สิงโต นำโชค</Artist>
```

❖ สามารถเขียน Element ซ้อนกันได้

```
<address>
    <road>มิตรภาพ</road>
    <zipcode>40002</zipcode>
</address>
```

Attribute

- Attribute คือ คุณสมบัติของ Element ที่ประกอบด้วยชื่อและค่า
- Attribute จะถูกวางภายใน Element เสมอ

```
<element_name attribute="value"> … </element_name>
```

- ตัวอย่างของ Attribute เช่น

ชื่อของ Attribute คือ id

ค่าของ Attribute คือ 123

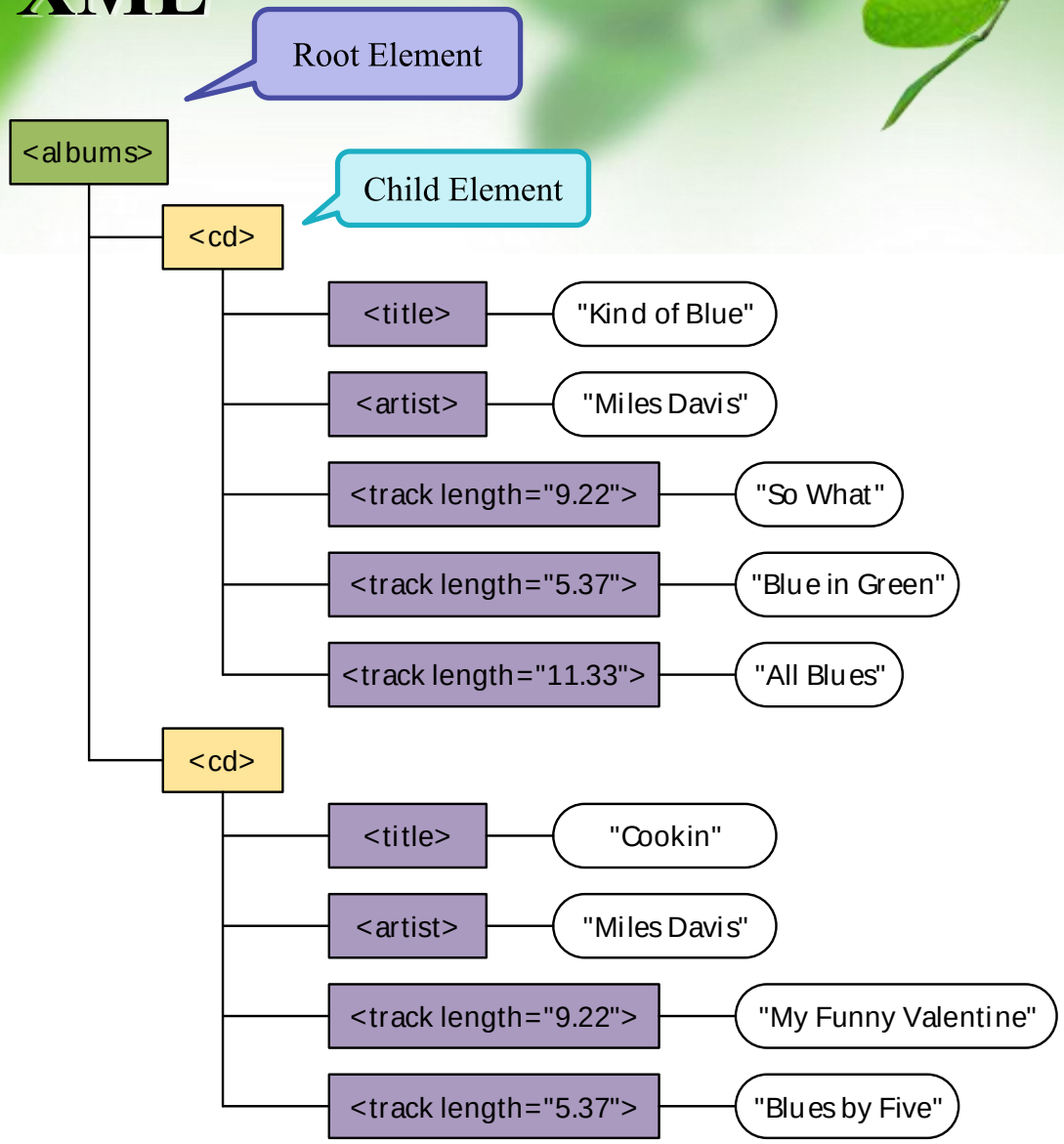
```
<employee  id="123">Somchai</employee>
```

```
<song name="อาย" artist="สิงโต นำโชค" />
```

ตัวอย่างโครงสร้าง XML

ข้อมูลอัลบั้มเพลง

CD TITLE	ARTIST	TRACKS
Kind of Blue	Miles Davis	So What (9.22)
		Blue in Green (5.37)
		All Blues (11.33)
Cookin	Miles Davis	My Funny Valentine (5.57)
		Blues by Five (9.53)



ตัวอย่างเอกสาร XML

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

<albums>
<cd>
   <title>Kind of Blue</title>
   <artist>Miles Davis</artist>
   <track length="9.22">So What</track>
   <track length="5.37">Blue in Green</track>
   <track length="11.33">All Blues</track>
</cd>
<cd>
   <title>Cookin</title>
   <artist>Miles Davis</artist>
   <track length="5.57">My Funny Valentine</track>
   <track length="9.53">Blues by Five</track>
</cd>
</albums>
```

กิจกรรม

❖ จากข้อมูลรายการสั่งซื้อสินค้าต่อไปนี้ จงออกแบบเอกสาร XML

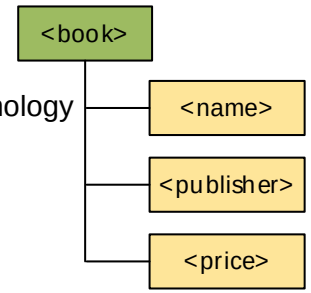
Order Date	Customer	Items order	Qty	Price
12/10/2015	Mr. Lee	A4 paper Ruler	5 2	120 25
	Mr. Som	Eraser	10	17
20/11/2015	Mr. Yue	Flash Drive A4 paper	2 3	130 120

ปัญหา Name Collision

❖ **Name collision** คือ การเกิดชื่อ Element หรือ Attribute ซ้ำกัน เมื่อมีการรวมข้อมูลที่อยู่ในรูปแบบ XML ตั้งแต่ 2 เอกสารขึ้นไป ทำให้เกิดความกำกวม

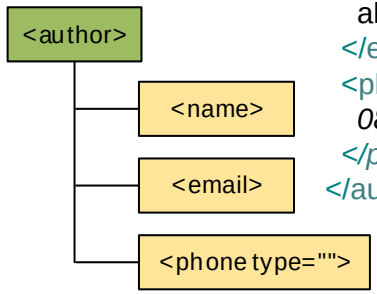
```
<book>
  <name>
    Web Services Technology
  </name>
  <publisher>
    ABC Printing
  </publisher>
  <price>250</price>
</book>
```

ข้อมูลหนังสือ

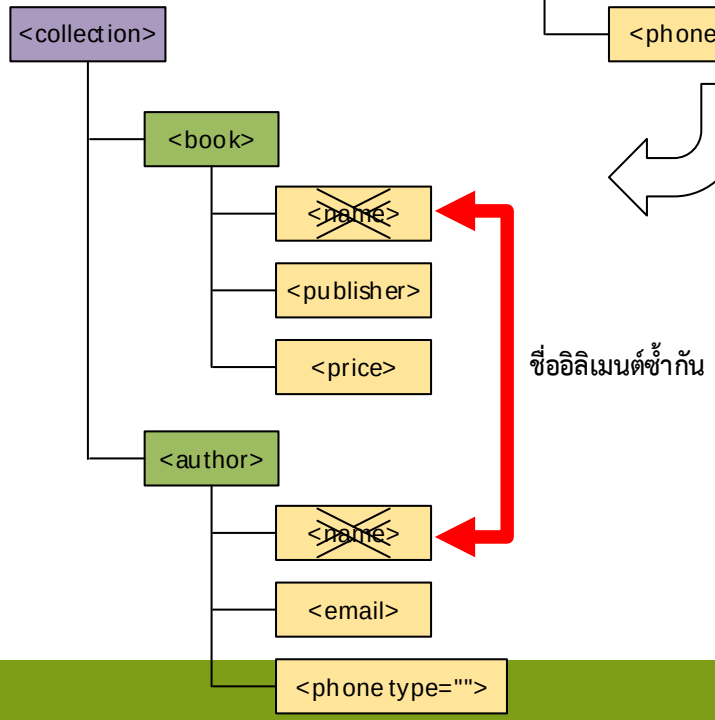



```
<author>
  <name>Albert M.</name>
  <email>
    albertm@gmail.com
  </email>
  <phone type="mobile">
    08 3543 3226
  </phone>
</author>
```

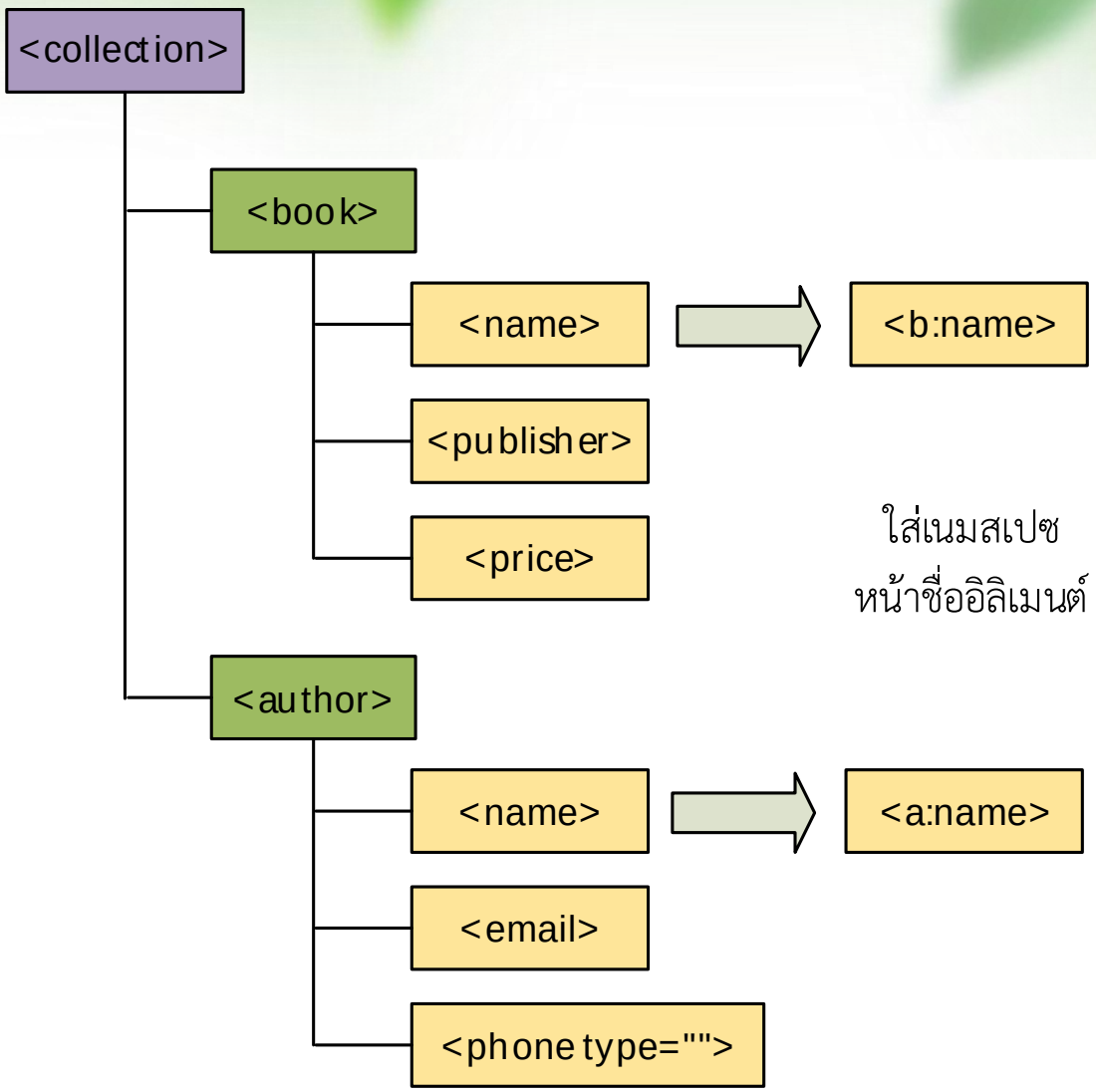
ข้อมูลผู้เขียน



ข้อมูลที่รวบรวมแล้ว

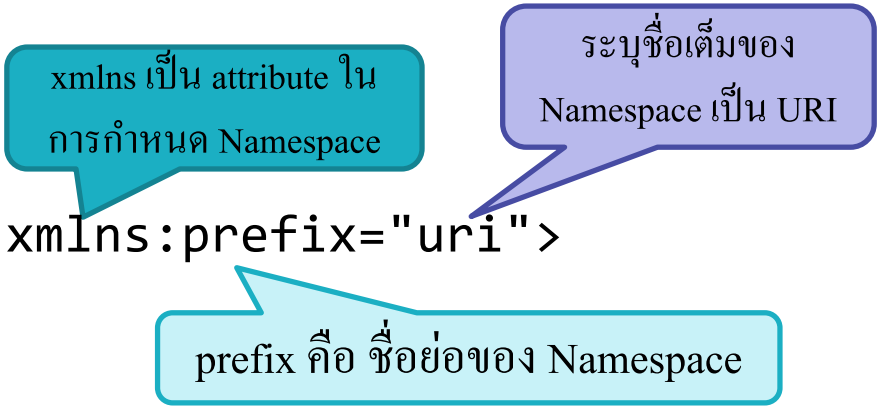


การแก้ปัญหา Name Collision



Namespace

- **Namespace** คือ ชื่อของกลุ่ม Element
- เมื่อระบุ Namespace ร่วมกับชื่อ Element แล้ว จะทำให้เกิดชื่อที่เป็นหนึ่งเดียว (Unique)
- Namespace ใช้ URI ในการกำหนดชื่อ แต่การใช้ URI ทำให้โค้ดยุ่งเหยิงจึงมีการระบุชื่อย่อ หรือ prefix
- รูปแบบการกำหนด Namespace



ตัวอย่าง

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

<collection>

  <b:book xmlns:b="http://book.org">
       <b:name>Web Services Technology</b:name>
       <b:publisher>ABC Printing</b:publisher>
       <b:price>250</b:price>
  </b:book>

  <a:author xmlns:a="http://person.com">
       <a:name>Albert M.</a:name>
       <a:email>albertm@gmail.com</a:email>
       <a:phone a:type="mobile">668 3543 3226</a:phone>
  </a:author>

</collection>
```

ชื่อย่อของ Namespace

ชื่อเต็มของ Namespace

Default Namespace

- Default Namespace ใช้ระบุ Namespace ที่ใช้ทั้งเอกสาร XML
- ทุกๆ Element และ Attribute ที่ไม่มีการกำหนด prefix จะถือว่าใช้ Default Namespace ทั้งหมด

ระบุด้วย Attribute xmlns ไม่ต้องใส่ prefix

```
<model xmlns="http://jacksonelect.com/models">
    <title>Laser4C (PR205)</title>
    <description>
       Entry level color laser printer
    </description>
    <type>color laser</type>
    <ordered>320</ordered>
    <parts list="chx201,fa100-5,eng005-2,cbx-450V4" />
</model>
```

ขอบเขตของ Namespace

- Namespace จะอยู่ภายใต้ที่ Element ที่ namespace นั้นถูกประกาศเริ่มตั้งแต่ Element เปิดไปจนถึง Element ปิด

Title อยู่ใน Namespace http://www.library.com

Title อยู่ใน Namespace http://www.otherlibrary.com

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

<Book xmlns="http://www.library.com">
   <Title>Sherlock Holmes - I</Title>
   <Author>Arthur Conan Doyle</Author>
   <Purchase xmlns="http://www.otherlibrary.com">
      <Title>Sherlock Holmes - II</Title>
      <Author>Arthur Conan Doyle</Author>
   </Purchase>
</Book>
```

กิจกรรม

❖ จากเอกสาร XML ต่อไปนี้ เกิด Name Collision จงแก้ปัญหาด้วยการเพิ่ม namespace

```
<?xml version="1.0" encoding="UTF-8"?>
<car>
  <body>
     <factory>oasis world</factory>
  </body>
  <wheel>
     <factory>bridgestone</factory>
  </wheel>
</car>
```

XPath

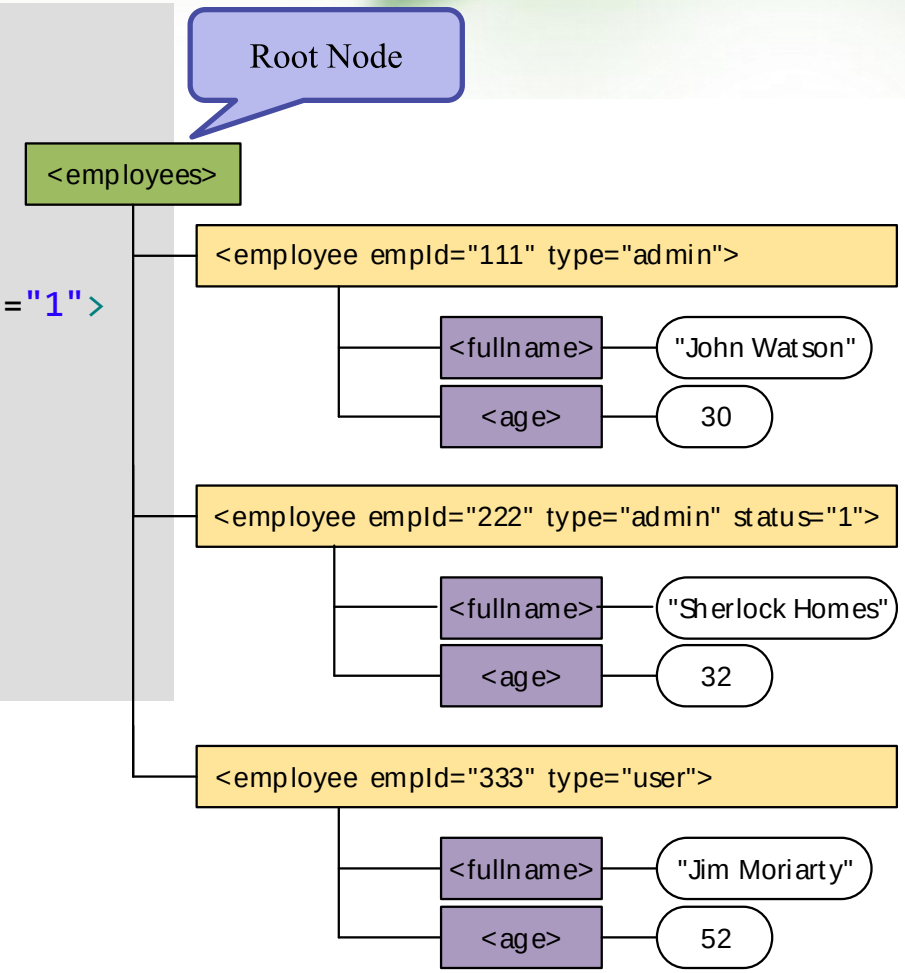
- XPath คือ ประโยคสัญลักษณ์ที่ใช้ในการดึงข้อมูลจากเอกสาร XML เพื่อนำไปประมวลผล หรือแสดงผล
- XPath กำหนดเส้นทางเพื่อไปยังโหนดต่างๆ บนเอกสาร XML
- หากเปรียบเทียบ XML เป็นฐานข้อมูล ภาษา XPath ก็เปรียบเสมือนภาษา SQL

สัญลักษณ์ของ XPath

รูปแบบ	ความหมาย
/	เลือก Node ที่ต้องการโดยเริ่มจาก Root Node
//	แทนเส้นทางของ Node ตั้งแต่ Root Node
.	เลือก Node ปัจจุบัน
..	เลือก Parent Node ของ Node ปัจจุบัน
@	เลือก attribute
nodename	เลือกทุก Node ที่มีชื่อเป็น "nodename"

ตัวอย่างเอกสาร XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<employees>
    <employee empId="111" type="admin">
        <fullname>John Watson</fullname>
        <age>30</age>
    </employee>
    <employee empId="222" type="admin" status="1">
        <fullname>Sherlock Homes</fullname>
        <age>32</age>
    </employee>
    <employee empId="333" type="user">
        <fullname>Jim Moriarty</fullname>
        <age>52</age>
    </employee>
</employees>
```

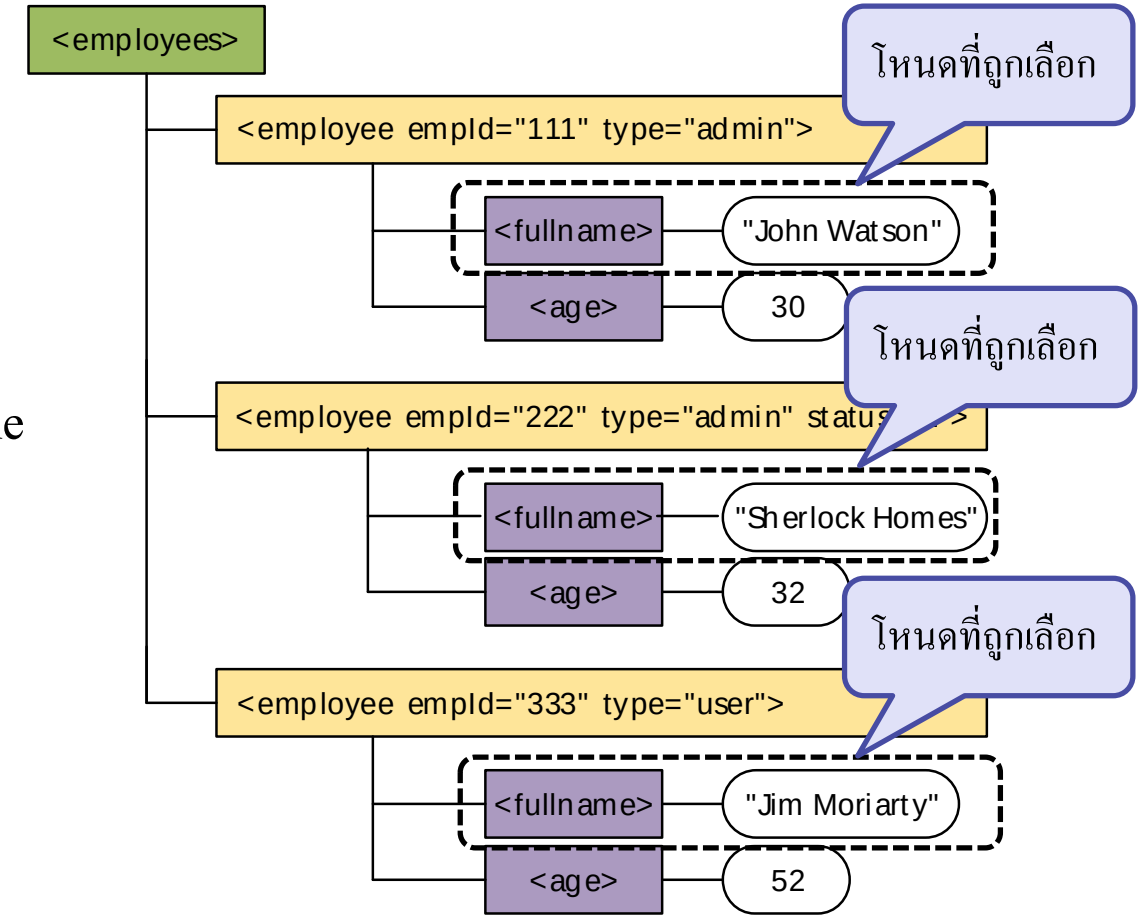


Absolute Path

- Absolute Path คือ การเลือกโหนดที่ต้องการ โดยระบุชื่อโหนดทั้งหมดตั้งแต่โหนด root

- ตัวอย่าง

/employees/employee/fullname

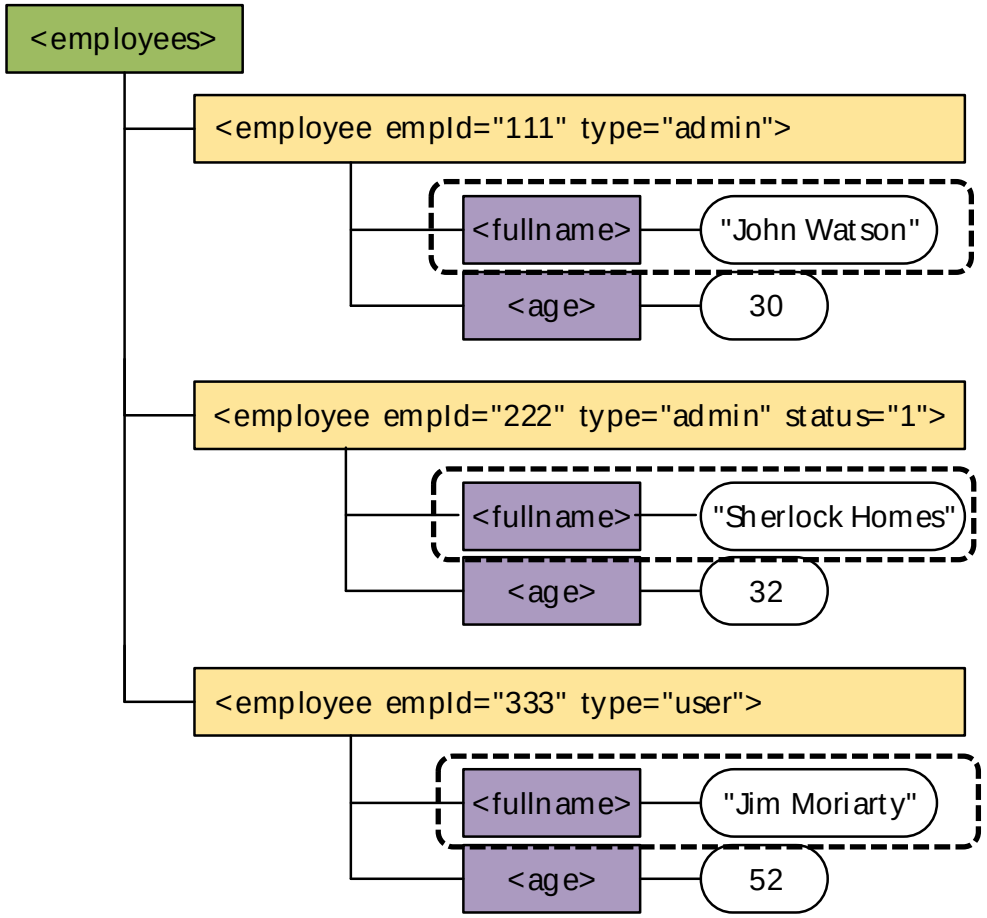


Relative Path

❖ Relative Path คือ การเลือกโหนดที่ต้องการ โดยละชื่อโหนดทั้งหมดด้วย //

❖ ตัวอย่าง

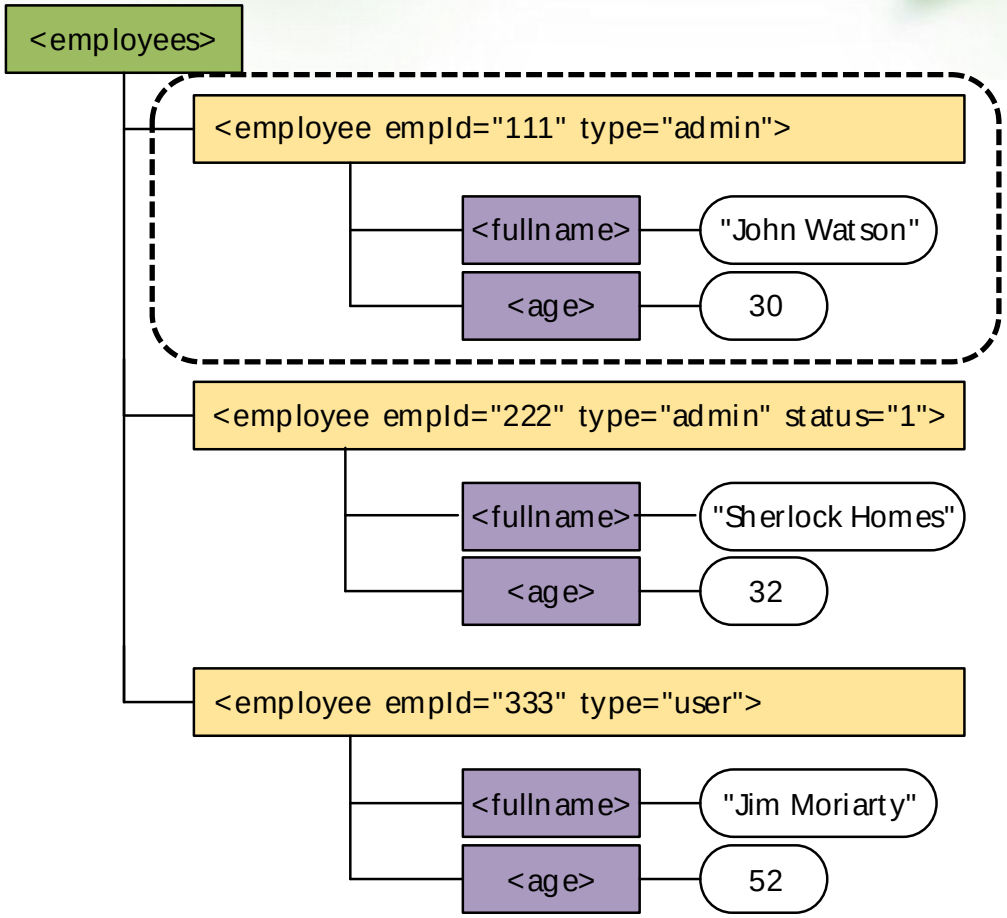
//fullname



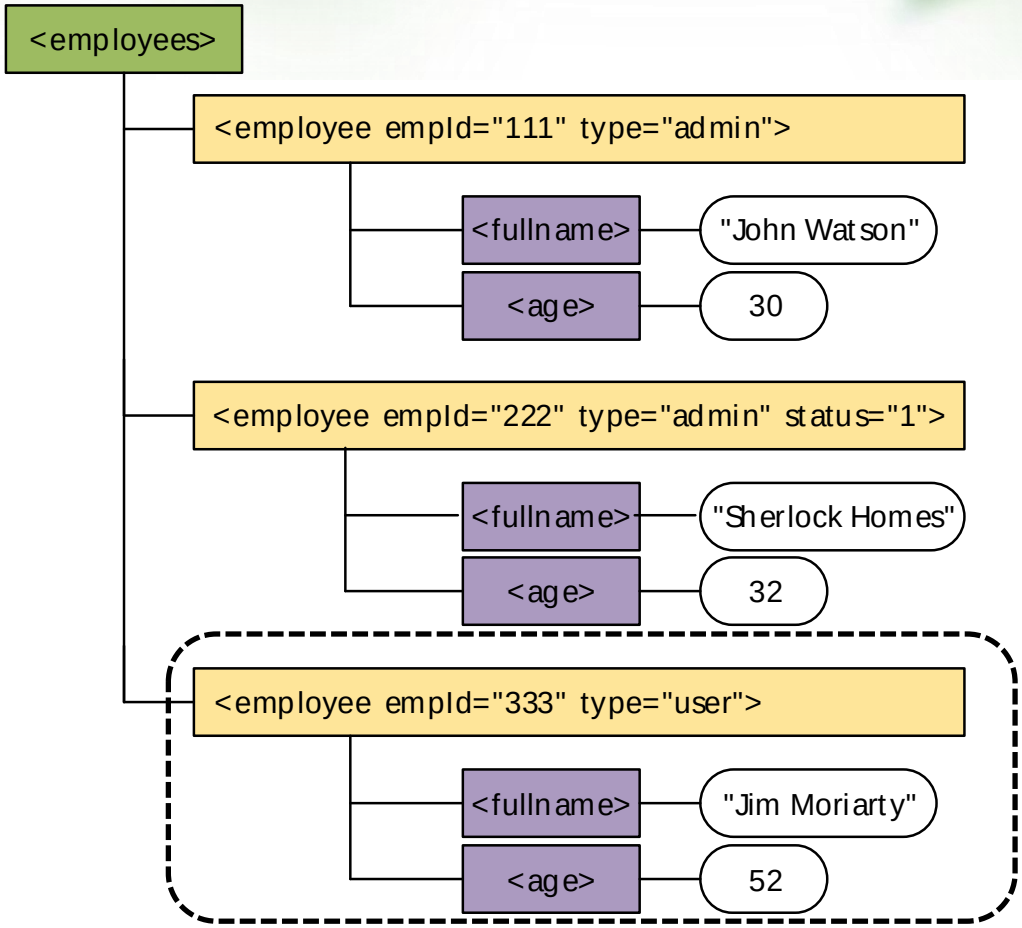
การเลือกแบบมีเงื่อนไข

รูปแบบ	ความหมาย
/employees/employee[1]	เลือกโหนด employee ซึ่งเป็นลูกของ employees เฉพาะโหนดแรก
/employees/employee[last()]	เลือกโหนด employee ซึ่งเป็นลูกของ employees เฉพาะโหนดสุดท้าย
/employees/employee[last()-1]	เลือกโหนด employee ซึ่งเป็นลูกของ employees เฉพาะโหนดรองสุดท้าย
//employee[@status]	เลือกโหนด employee ที่มี attribute ชื่อ status
//employee[@type='admin']	เลือกโหนด employee ที่มี attribute ชื่อ type และมีค่าเป็น "admin"
//employee[@type='admin']/fullname	เลือกโหนด fullname จากโหนด employee ที่มี attribute ชื่อ type และมีค่าเป็น "admin"
/employees/employee[age>40]/fullname	เลือกโหนด fullname จากโหนด employee ที่มีโหนดลูกชื่อ age มีค่ามากกว่า 40
/employees/employee[position() <= 2]/fullname	เลือกโหนด fullname จากโหนด employee ที่อยู่ตำแหน่ง 1 และ 2
//employee/@type	เลือกข้อมูลใน attribute ชื่อ type ของโหนด employee

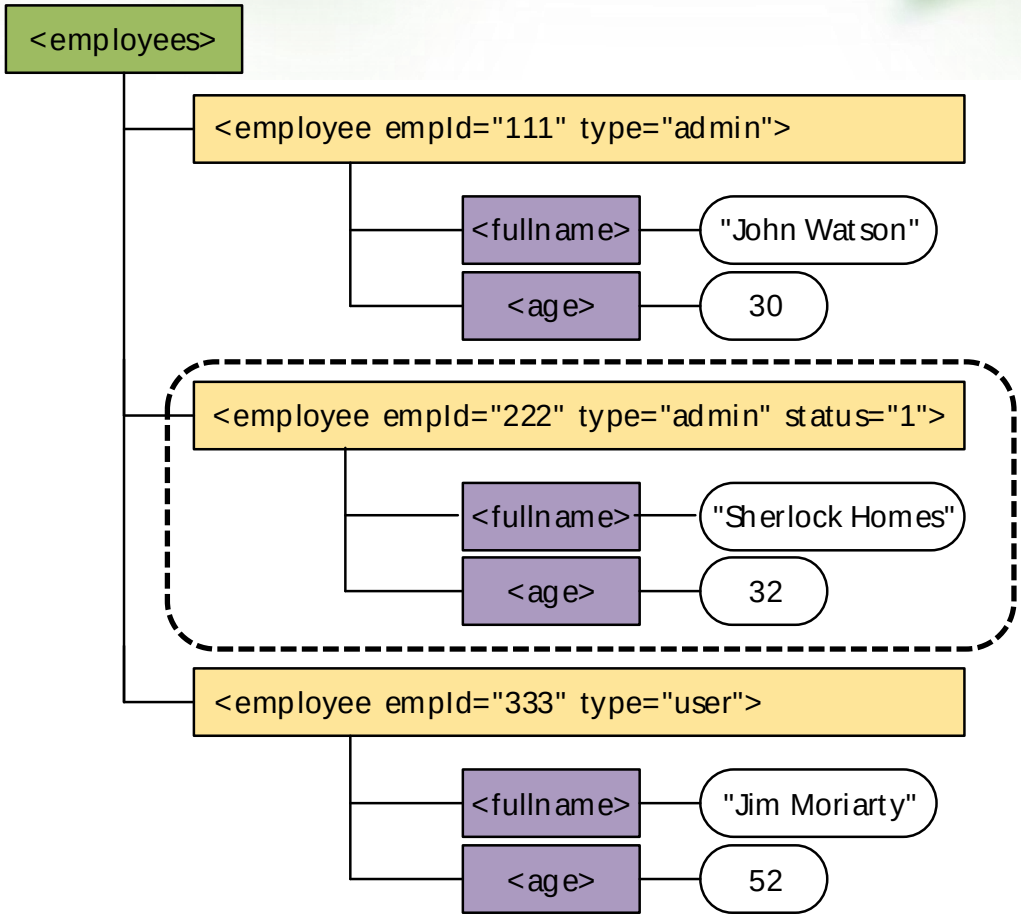
/employees/employee[1]



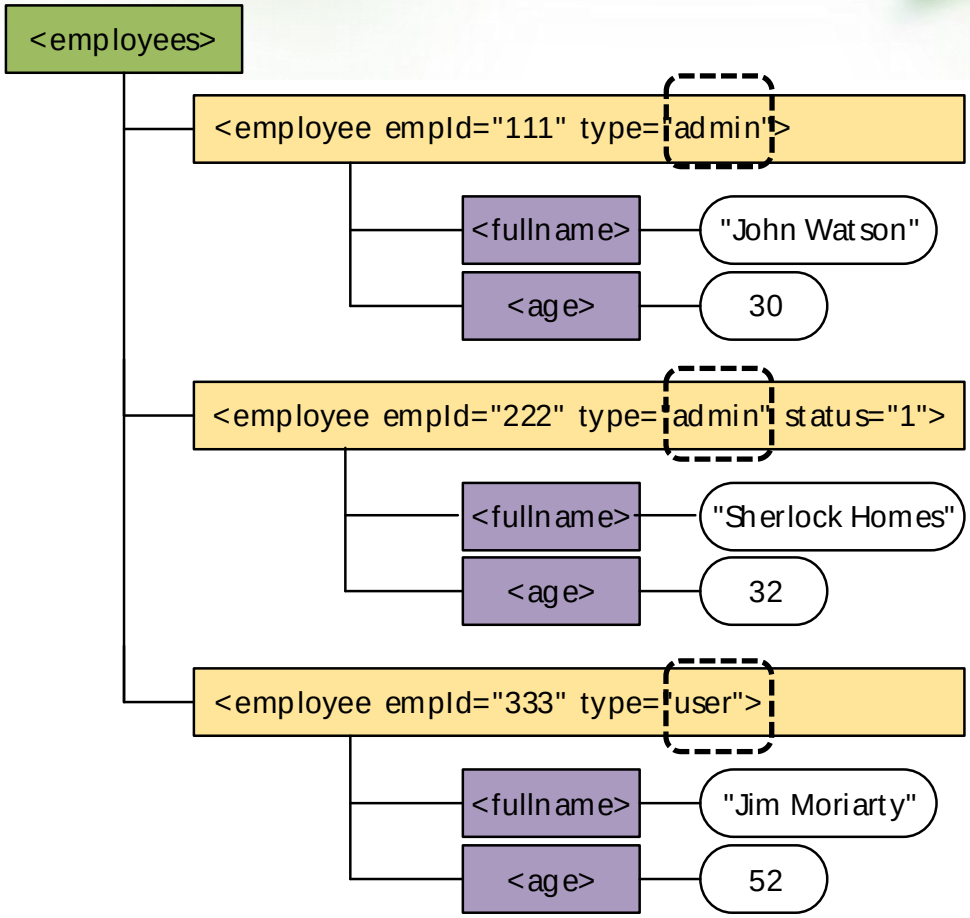
/employees/employee[last()]



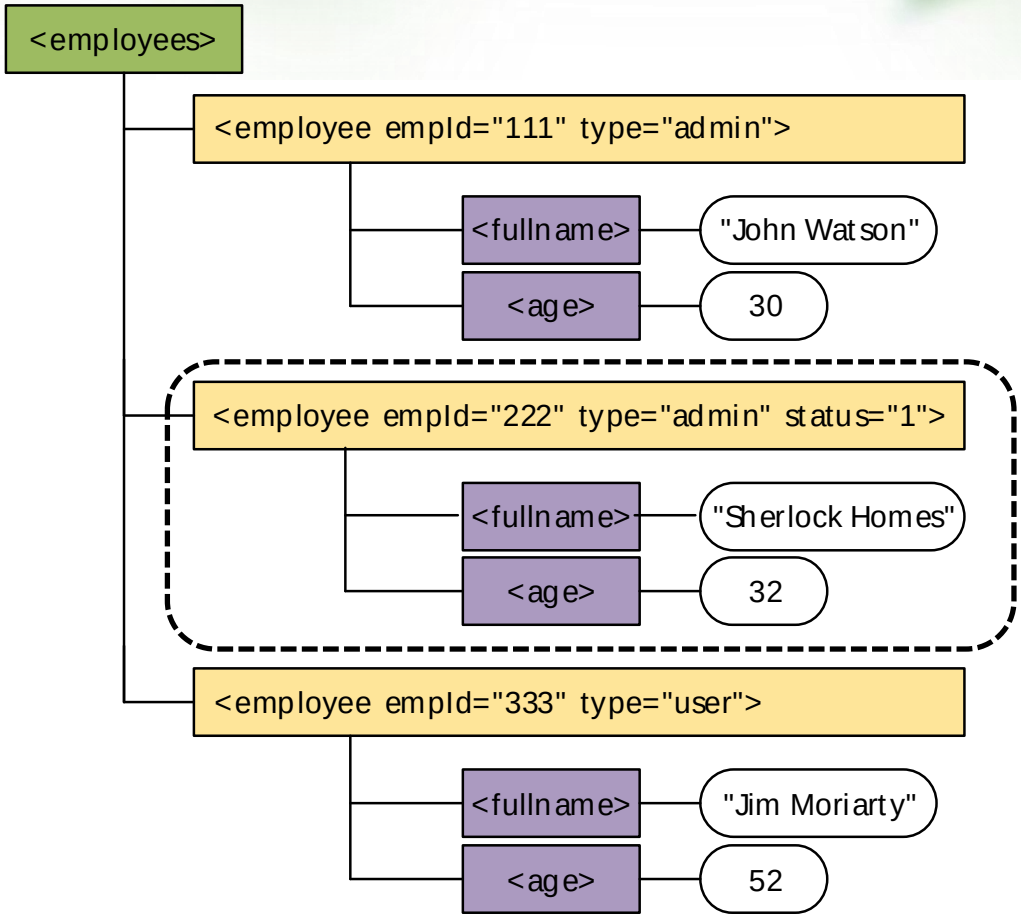
/employees/employee[last()-1]



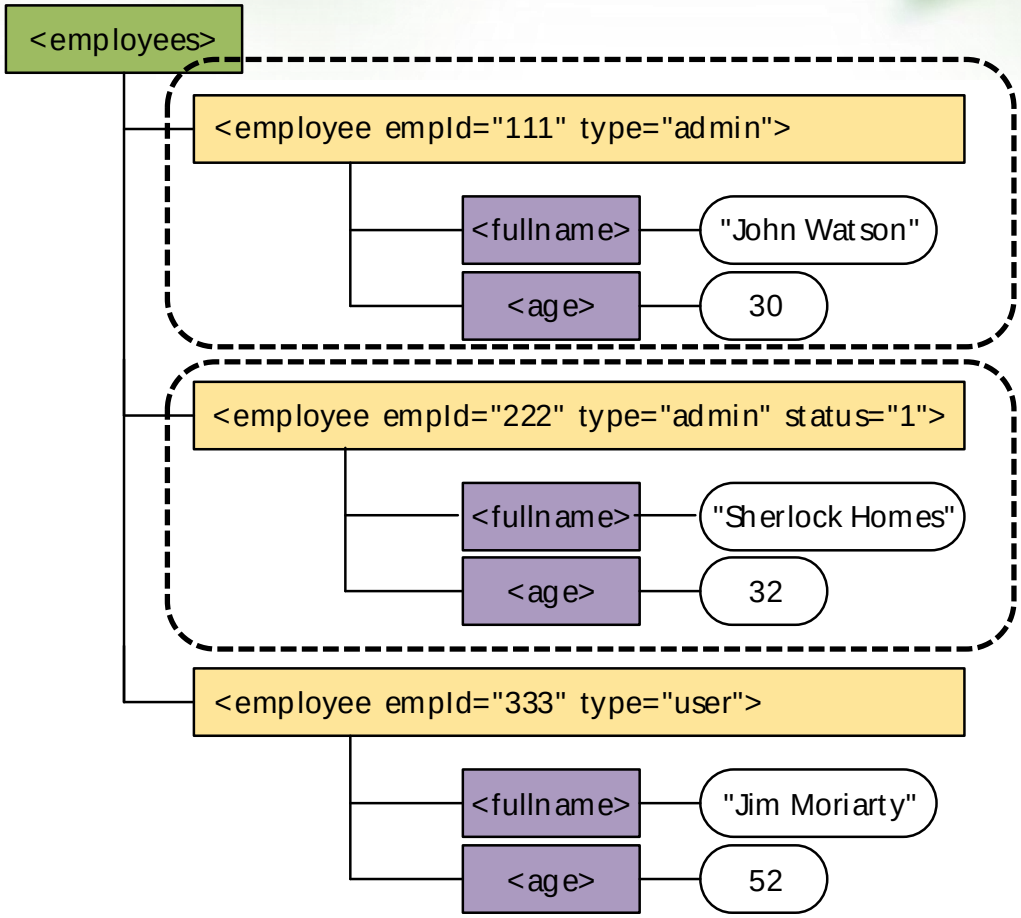
//employee/@type



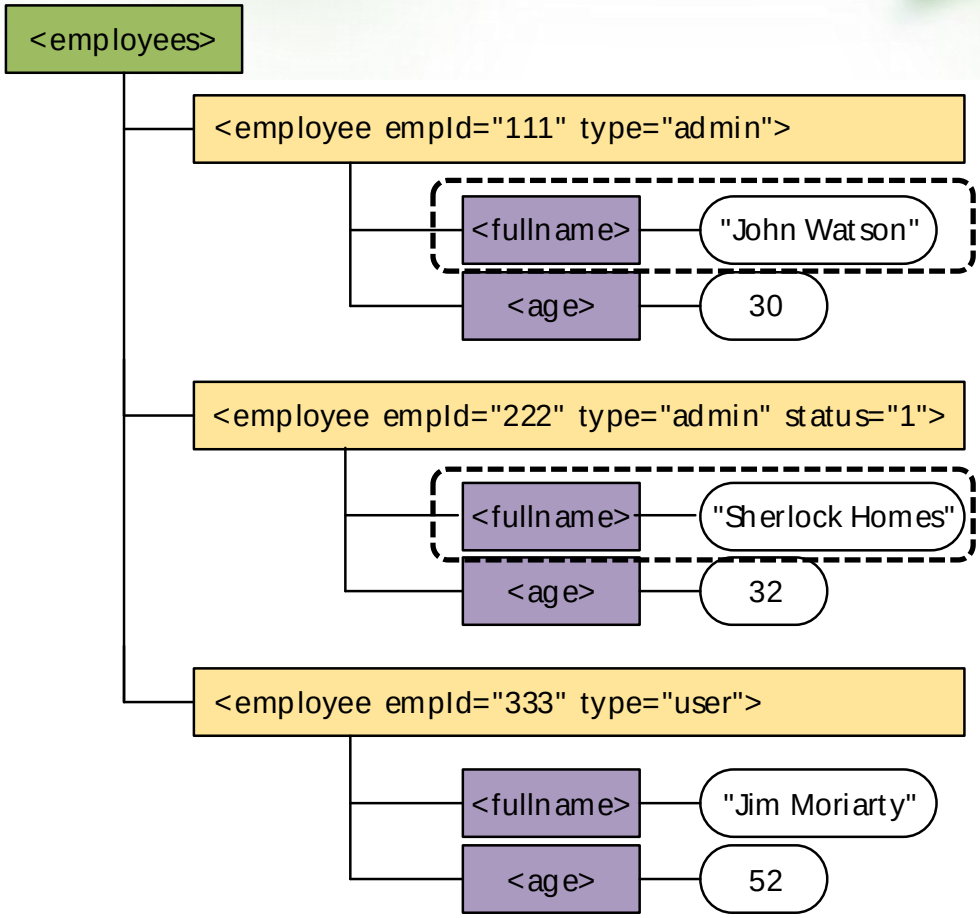
//employee[@status]



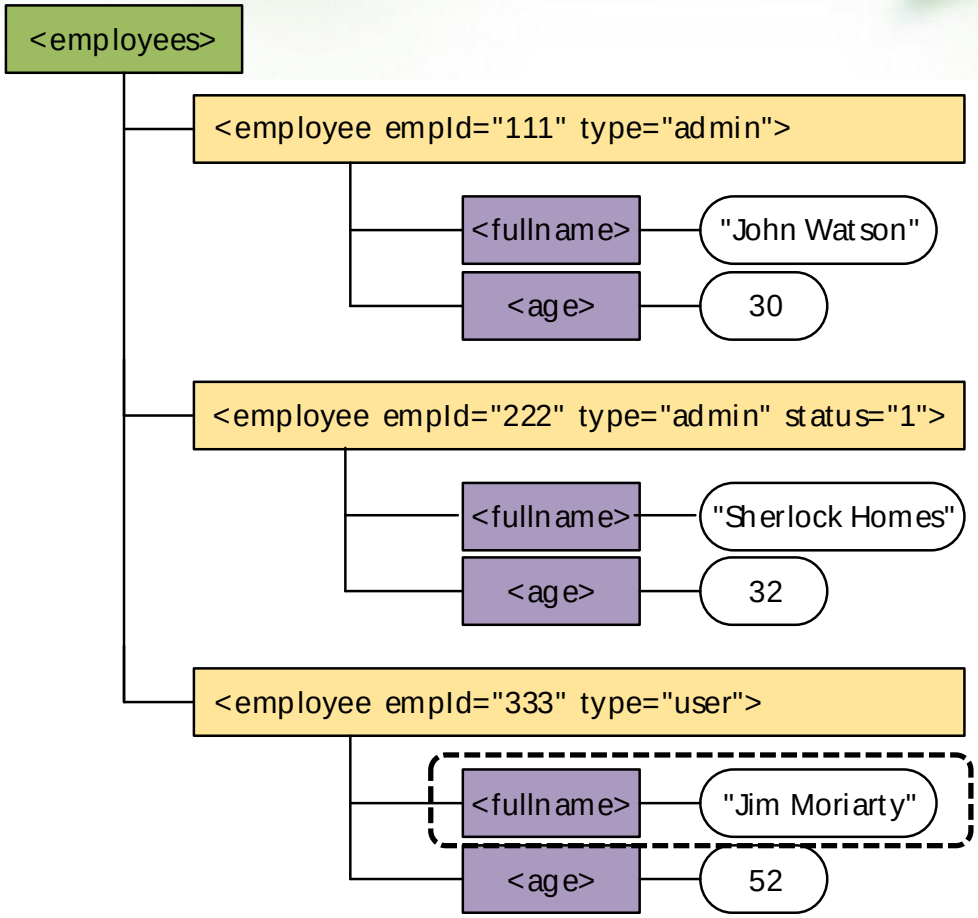
//employee[@type='admin']



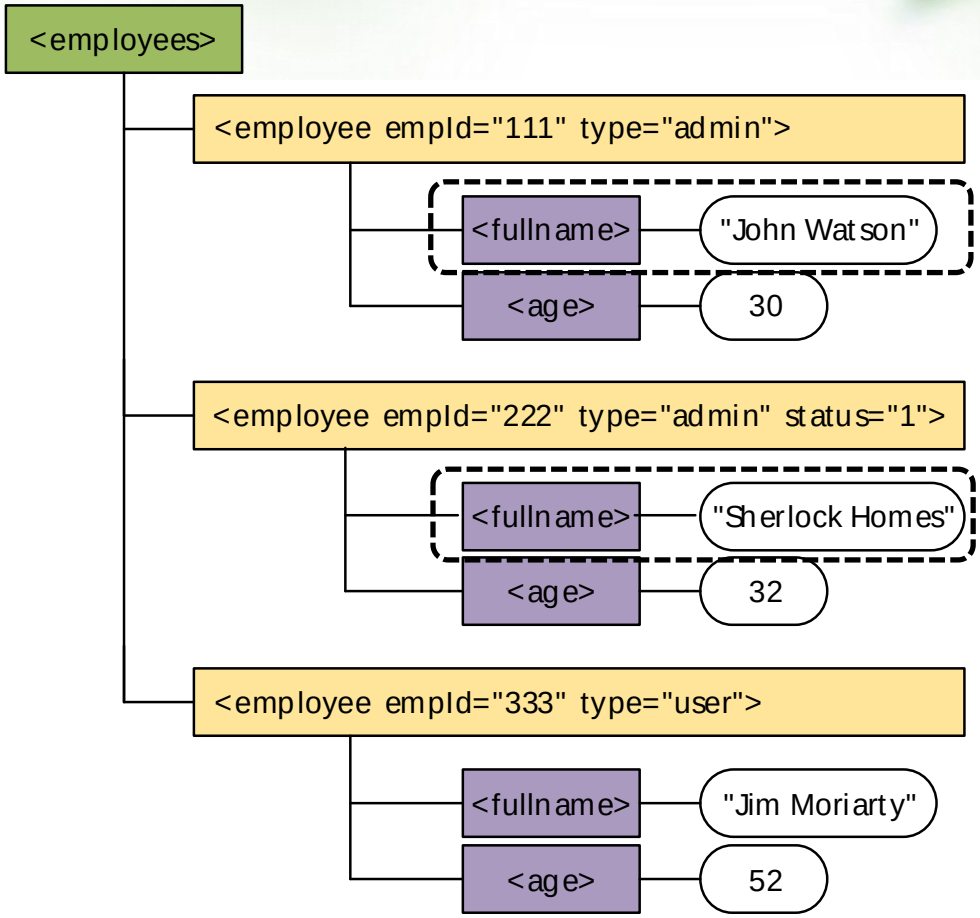
//employee[@type='admin']/fullname



/employees/employee[age>40]/fullname



/employees/employee[position()<=2]/fullname

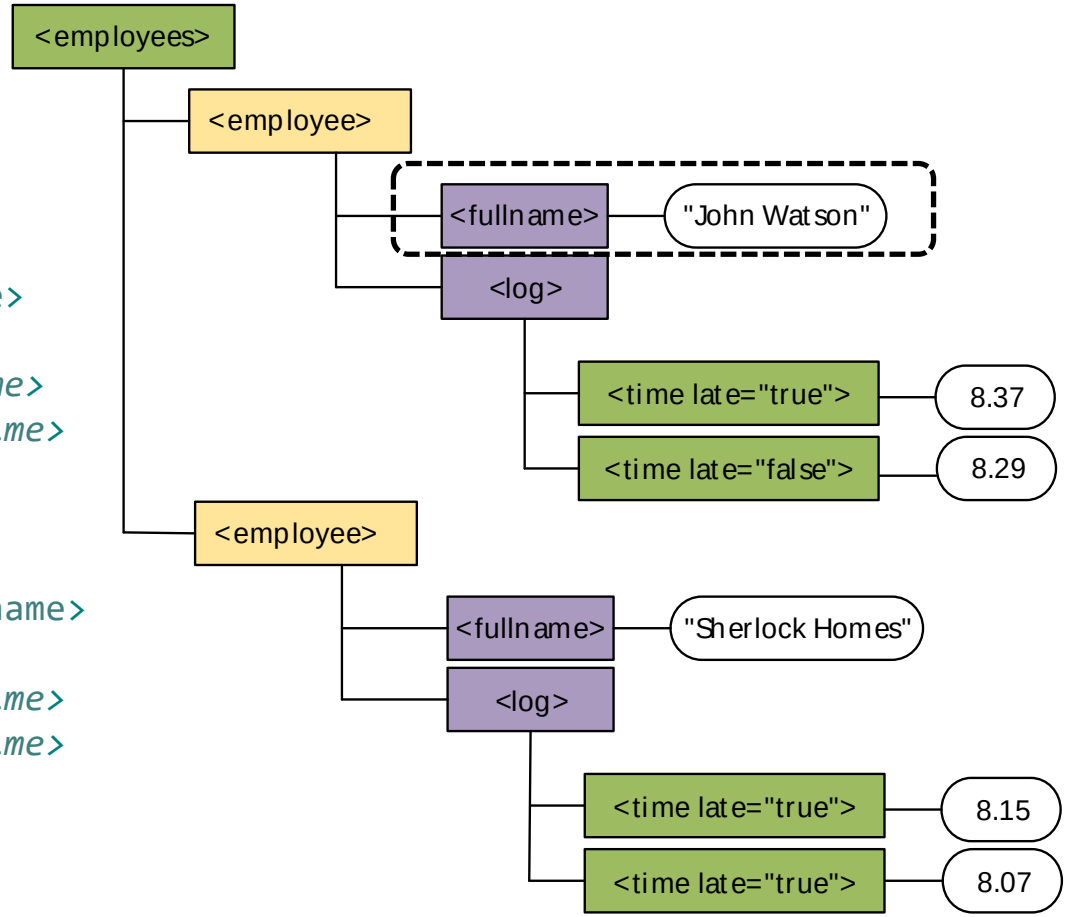


การใช้ ..

ต้องการชื่อพนักงานที่มี attribute late เป็น true

//time[@late='true']/../../fullname

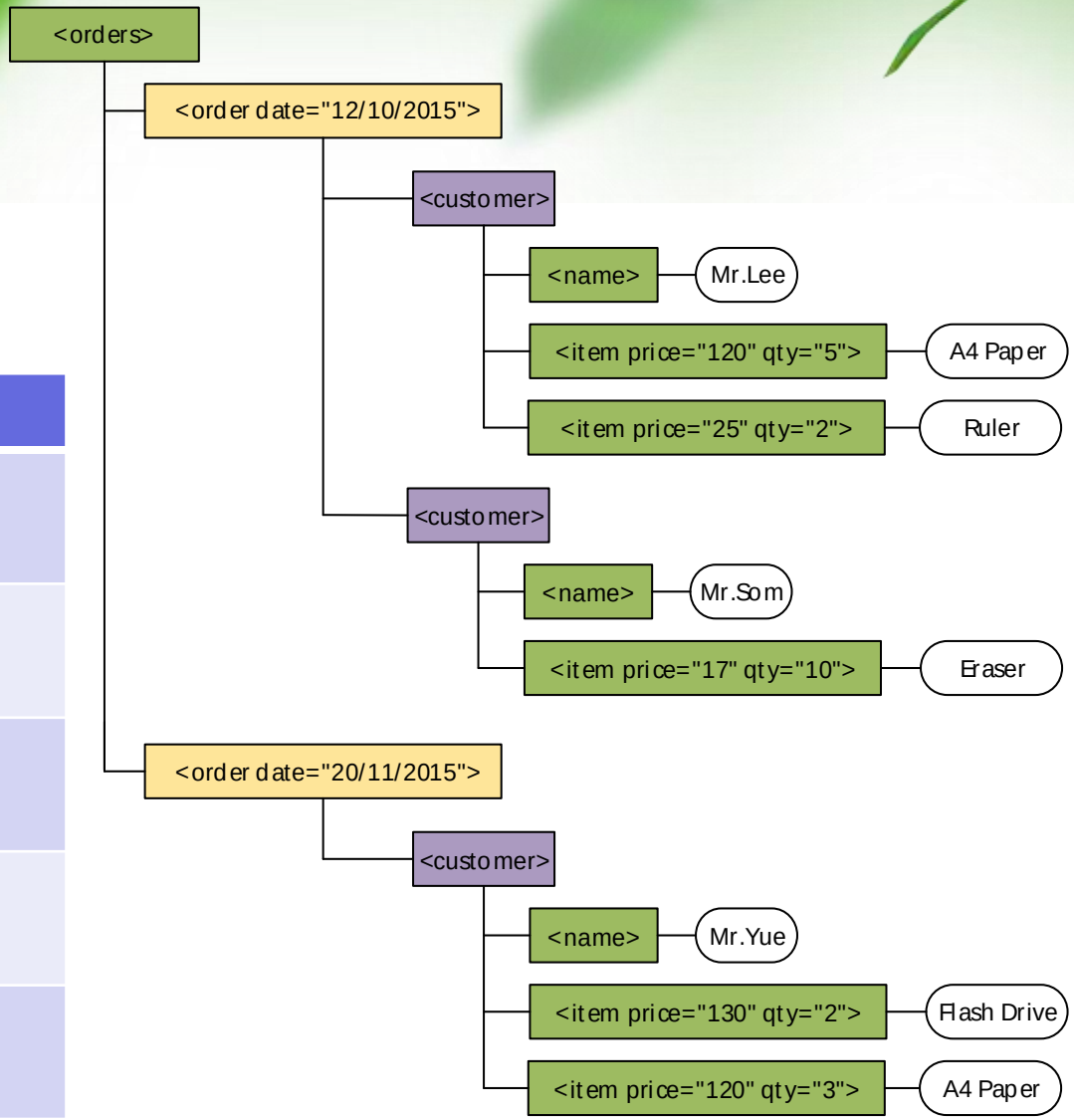
```xml
<?xml version="1.0" encoding="UTF-8"?>
<employees>
    <employee>
        <fullname>John Watson</fullname>
        <log>
            <time late="true">8.37</time>
            <time late="false">8.29</time>
        </log>
    </employee>
    <employee>
        <fullname>Sherlock Homes</fullname>
        <log>
            <time late="false">8.15</time>
            <time late="false">8.07</time>
        </log>
    </employee>
</employees>
```



กิจกรรม

❖ จงเขียน XPath สำหรับดึงข้อมูลจากเอกสาร XML ซึ่งมีโครงสร้างและข้อมูลดังภาพ

เงื่อนไข	XPath
แสดงชื่อลูกค้าทั้งหมด	
แสดงชื่อสินค้าที่มีราคามากกว่า 100 บาท	
แสดงชื่อลูกค้าที่ซื้อของในวันที่ 12/10/2015	
แสดงชื่อลูกค้าที่ซื้อ 'A4 Paper'	
แสดงชื่อลูกค้าที่ซื้อสินค้าที่มีราคามากกว่า 125 บาท	



```
<?xml version="1.0" encoding="UTF-8"?>
<orders>
  <order date="12/10/2015">
  <customer>
     <name>Mr.Lee</name>
     <item qty="5" price="120">A4 paper</item>
     <item qty="2" price="25">Ruler</item>
  </customer>
  <customer>
     <name>Mr.Som</name>
     <item qty="10" price="17">Eraser</item>
  </customer>
</order>
<order date="20/11/2015">
   <customer>
      <name>Mr. Yue</name>
      <item qty="2" price="130">Flash Drive</item>
      <item qty="3" price="120">A4 Paper</item>
   </customer>
</order>
</orders>
```

การใช้ XPath ดึงข้อมูลจากเอกสาร XML โดยใช้ภาษา Java

1. อ่านข้อมูลจากไฟล์ XML เก็บลง document object

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document document = builder.parse(new File("D:/employee.xml"));
```

2. สร้าง object สำหรับแปลคำสั่ง XPath

```
XPath xPath = XPathFactory.newInstance().newXPath();
```

3. ดึงข้อมูลด้วย XPath

```
String result = xPath.compile("//employee[2]/fullname").evaluate(document);
```

ผลลัพธ์จากการแปลคำสั่ง XPath

ผลลัพธ์จากการแปลคำสั่ง XPath มี 3 ชนิด

- String – เป็นข้อความ ซึ่งหากเป็นตัวเลข ที่ต้องการนำไปแสดงผลจะต้องแปลงชนิดข้อมูลก่อน
- Node – เป็น Object ของ 1 Element
- NodeList – เป็นชุดของ Object ของตั้งแต่ 1 Element ขึ้นไป ต้องวนลูปเพื่อเข้าถึง

การใช้ XPath ดึงข้อมูลจากไฟล์ XML

```
import java.io.File;
import javax.xml.parsers.*;
import javax.xml.xpath.*;
import org.w3c.dom.*;
public class XPathApplication {
  public static void main(String[] args) throws Exception {
     // 1. อ่านข้อมูลจากไฟล์ XML เก็บลง document object
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document document = builder.parse(new File("D:/employee.xml"));

    // 2. สร้าง object สำหรับแปลคำสั่ง XPath
    XPath xPath = XPathFactory.newInstance().newXPath();

    // 3. ดึงข้อมูลด้วย  XPath
    // 3.1 กรณีที่ผลลัพธ์มีค่าเดียว
    String result = xPath.compile("//employee[2]/fullname").evaluate(document);
    System.out.println("ชื่อพนักงานคนที่ 2 คือ " + result + "\n");

    // 3.2 กรณีที่ผลลัพธ์มีหลายค่า
    System.out.println("รายชื่อพนักงานทั้งหมด");
    NodeList nodeList = (NodeList) xPath.compile("//fullname").evaluate(document,
                                                    XPathConstants.NODESET);
    for (int i=0; i<nodeList.getLength(); i++) { // วนลูปแสดงข้อมูลที่ได้จาก XML
        System.out.println(nodeList.item(i).getFirstChild().getNodeValue());
    }
  }
}
```

ผลลัพธ์

```
ชื่อพนักงานคนที่ 2 คือ Sherlock Homes
รายชื่อพนักงานทั้งหมด
John Watson
Sherlock Homes
Jim Moriarty
```

การเรียกใช้ Web API

- ศึกษารายละเอียดการเรียกใช้จากเอกสารของผู้ให้บริการ
  - URL
  - ค่าที่ต้องส่ง
- สมัครสมาชิก ซึ่งบางเว็บไม่จำเป็น
- เขียนโปรแกรม
  - สร้างคลาสเก็บข้อมูลจากเว็บเซอร์วิส
  - เขียนชุดคำสั่งในการเรียกใช้ และดึงข้อมูลที่ได้ไปแสดงผล

Google Geocoding API

https://developers.google.com/maps/documentation/geocoding/

Geocoding API Request Format

A Geocoding API request must be of the following form:

```
https://maps.googleapis.com/maps/api/geocode/output?parameters
```

where `output` may be either of the following values:

- `json` (recommended) indicates output in JavaScript Object Notation (JSON)
- `xml` indicates output as XML

To access the Geocoding API over HTTP, use:

```
http://maps.googleapis.com/maps/api/geocode/output?parameters
```

HTTP is not recommended for applications that include sensitive user data, such as a user's location, in requests.

Some parameters are required while some are optional. As is standard in URLs, parameters are separated using the ampersand (&) character.

Google Maps API for Work users must include valid `client` and `signature` parameters with their Geocoding requests. Please refer to Google Maps API for Work Web Services for more information.

The rest of this page describes geocoding and reverse geocoding separately, because different parameters are available for each type of request.

Geocoding (Latitude/Longitude Lookup)

Required parameters in a geocoding request:

- `address` – The street address that you want to geocode, in the format used by the national postal service of the country concerned. Additional address elements such as business names and unit, suite or floor numbers should be avoided. Please refer to the FAQ for additional guidance.

  or

  `components` – A component filter for which you wish to obtain a geocode. See Component Filtering for more information. The components filter will also be accepted as an optional parameter if an `address` is provided.

Google Geocoding API

Server: https://maps.google.com

Resource:

/maps

/api

/geocode

/xml

?address=[ชื่อสถานที่]

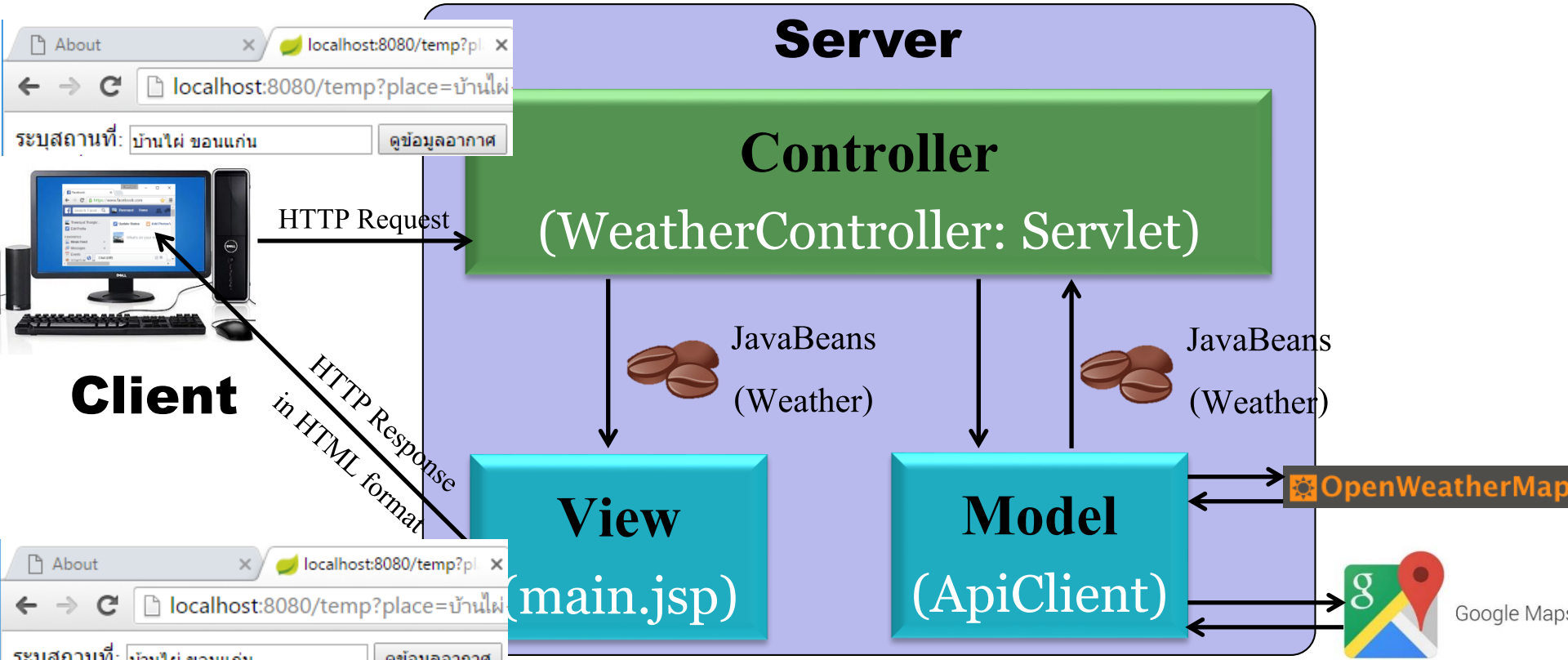
https://maps.googleapis.com/maps/api/geocode/xml?address=ขอนแก่น

การใช้ XPath ดึงข้อมูลจาก Web API

```
OkHttpClient client = new OkHttpClient();

// 1. กำหนด URL ของเว็บเซอร์วิส Google
String googleUrl = "https://maps.googleapis.com/maps/api/geocode/xml?address=" + placeName;

// 2. เรียกใช้ Web API
Request request = new Request.Builder().url(googleUrl).build();
Response response = client.newCall(request).execute();

// 3. แปลงข้อมูลจาก String ให้อยู่ในรูปแบบ Object ของ XML
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document document = builder.parse(response.body().byteStream());

// 4. สร้าง object สำหรับแปลคำสั่ง XPath
XPath xPath = XPathFactory.newInstance().newXPath();

// 5. ดึงข้อมูลด้วย XPath
String lat = xPath.compile("//location/lat").evaluate(document);
String lng = xPath.compile("//location/lng").evaluate(document);
```

Assignment#6

- สร้างเว็บไซต์แสดงอุณหภูมิของอากาศ ณ ปัจจุบัน ตามชื่อสถานที่ที่ผู้ใช้กรอก ตามหลัก MVC โดยให้ส่วน Model ทำหน้าที่ขอข้อมูล XML ที่ตั้งและอุณหภูมิจาก Google Geocoding API และ OpenWeatherMap API ซึ่งมีรายละเอียดดังคู่มือ http://openweathermap.org/current

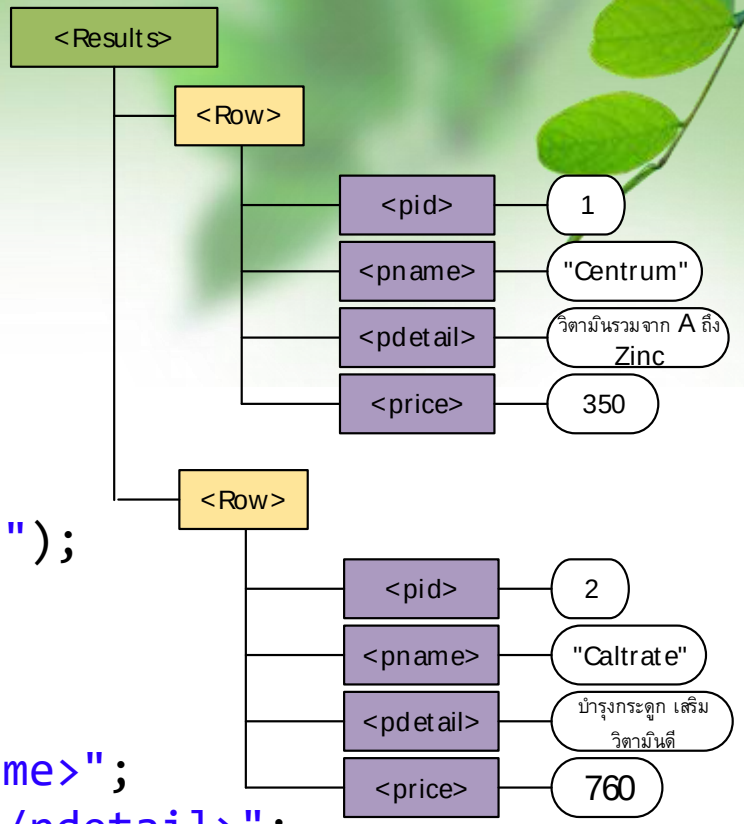


การสร้าง Web API

- Web API หรือ Web Services Provider คือ การเปิดข้อมูลให้กับระบบอื่นที่ต้องการใช้ข้อมูลร่วมกัน แทนการให้นักพัฒนาเข้าถึงฐานข้อมูลโดยตรง นอกจากนี้ยังช่วยให้ระบบที่พัฒนาด้วยเทคโนโลยีที่แตกต่างกันสามารถทำงานร่วมกันได้
- การนำข้อมูลเปิดให้บริการในรูปแบบ Web API จะต้องแปลงข้อมูลให้อยู่ในรูปแบบมาตรฐานก่อน เช่น XML หรือ JSON ในที่นี่จะกล่าวถึงรูปแบบ XML ซึ่งสามารถทำได้ 2 วิธี
  - **แบบที่ 1:** ดึงผลลัพธ์ที่ได้จากการประมวลผล และใส่แท็ก XML ระหว่างเข้าถึงข้อมูลแต่ละ record
  - **แบบที่ 2:** ดึงผลลัพธ์ที่ได้จากการประมวลผล และเก็บลง Object Document ซึ่งเป็น Object ที่เก็บข้อมูลในรูปแบบโครงสร้างของ XML

แบบที่ 1

```
String xml = "<Results>";

while (rs.next()) {
    int pid = rs.getInt("pid");
    String pname = rs.getString("pname");
    String pdetail = rs.getString("pdetail");
    int price = rs.getInt("price");
    xml = xml + "<Row>";
    xml = xml + "<pid>" + pid + "</pid>";
    xml = xml + "<pname>" + pname + "</pname>";
    xml = xml + "<pdetail>" + pdetail + "</pdetail>";
    xml = xml + "<price>" + price + "</price>";
    xml = xml + "</Row>";
}
xml = xml + "</Results>";
System.out.println(xml);
```



ผลลัพธ์

```
<Results><Row><pid>1</pid><pname>Centrum</pname><pdetail>วิตามินรวมจาก A ถึง Zinc</pdetail><price>350</price></Row><Row><pid>2</pid><pname>Caltrate</pname><pdetail>บำรุงกระดูก เสริมวิตามินดี</pdetail><price>760</price></Row><Row><pid>3</pid>…</Row></Results>
```

แบบที่ 2

```
// ดึงข้อมูลของผลลัพธ์ที่ได้จากฐานข้อมูล
ResultSetMetaData rsmd = rs.getMetaData();
int colCount = rsmd.getColumnCount();

// สร้าง  Object ของเอกสาร XML สำหรับเก็บผลลัพธ์
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document doc = builder.newDocument();
Element results = doc.createElement("Results");
doc.appendChild(results);

while (rs.next()) {
    Element row = doc.createElement("Row");
    results.appendChild(row);
    for (int i = 1; i <= colCount; i++) {
        String columnName = rsmd.getColumnName(i);
        String value = rs.getString(i);
        Element node = doc.createElement(columnName);
        node.appendChild(doc.createTextNode(value));
        row.appendChild(node);
    }
}
```

กิจกรรม

- สร้าง Web API ที่ให้เฉพาะข้อมูลบางส่วนจากฐานข้อมูล ได้แก่ firstname, lastname, dept_name
- สร้างฐานข้อมูลใหม่ชื่อ xcompany เลือก Collation เป็น utf8_general_ci
- ใช้ไฟล์ xcompany.sql ในการสร้างตารางและเพิ่มข้อมูล
- ตาราง department และ employee มีโครงสร้างและข้อมูลดังนี้

ตาราง department

dept_id	dept_name
1	ฝ่ายสารสนเทศ
2	ฝ่ายทรัพยากรบุคคล
3	ฝ่ายบัญชี

ตาราง employee

emp_id	firstname	lastname	position	dept_id
1	บุญมี	มากเหลือ	โปรแกรมเมอร์	1
2	สมัย	สมภาร	นักบัญชี	3
3	บุญช่วย	กาโว	ผู้จัดการ	2
4	สมใจ	บุญหลาย	Tester	1
5	ยี่หวา	เร่าร้อน	ธุรการ	2