0 0
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
9 9

Langages pour le Web

L3 MIAGE
17 mai 2018

Durée: 1h30
Aucun document autorisé
Barême sur 15 points.

Toutes les réponses doivent être données sur le sujet.

Ne pas mettre son nom sur le sujet, ni son numéro d'étudiant à gauche (place réservée au numéro de copie).

La mairie de Villard de Lans, petite ville du massif du Vercors et station de sports toutes saisons, souhaite mettre en place un flux `RSS` informant sur la météorologie, à raison de uniquement deux relevés par jour, précisément un a 6h00 du matin et l'autre à 12h00. Ces flux d'informations doivent renseigner sur 3 points : (1) *la température* incluant la mesure de température (une température est une valeur supérieure à -273°C) et éventuellement la valeur de l'isotherme zéro (altitude en m à laquelle il fait 0°C, cette mesure n'étant pas nécessairement fournie), (2) *les précipitations* incluant, s'il y a eu des précipitations, le niveau de précipitations en *mm* d'eau et le type de précipitation (parmi les valeurs : *aucune*, *pluie*, *grésil*, *grêle*, *neige*), et (3) *l'anémométrie*, c'est à dire la vitesse du vent en *km/h*, avec éventuellement la vitesse des rafales de vent (cette mesure n'étant pas nécessairement fournie), et la direction du vent avec 8 directions possibles (*N*, *NO*, *O*, *SO*, *S*, *SE*, *E*, *NE*), cette information n'étant pas nécessairement fournie (par exemple lorsqu'il n'y a pas de vent). Les données sont stockées au format `XML`, selon un schéma suffisamment générique pour être par la suite étendu à la communauté de communes de ce massif montagneux. Un fichier par mois est produit et est nommé `meteo-ville-mmyyyy.xml` (par exemple `meteo-VillarddeLans-122017.xml`). En plus des bulletins météo, chaque fichier contient le renseignement sur la période de l'année (mois et année), ainsi que les informations sur le site sur lequel sont effectuées les mesures. Les informations géographiques sont au format `KML`. NB: Tous les renseignements nécessaires à la modélisation se trouvent dans ce texte !

1 Modélisation des données météo en `XML` et `XMLSchema` (10 points)

Dans cette partie, nous allons modéliser les données météorologiques convenablement. On fourni ci-dessous une instance de fichier XML tel que souhaité pour le mois de décembre ; on ne montre ici que les bulletins du 18 décembre.
→ Fichier ***meteo-VillarddeLans-122017.xml*** :

```
<?xml version="1.0" encoding="UTF-8"?>
<met:météo
    xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xmlns:kml='http://www.opengis.net/kml/2.2'
    xmlns:gx="http://www.google.com/kml/ext/2.2"
    xmlns:met='http://www.meteo-france.fr/bulletin'
    xsi:schemaLocation='http://www.meteo-france.fr/bulletin ./meteo.xsd'
    url="http://www.espace-villard-correncon.fr/meteo-villard-de-lans.htm">
    <met:période année="2017" mois="12"/>
    <met:lieu>
        <kml:name>Villard de Lans</kml:name>
        <kml:description>Alpes Vercors Nord</kml:description>
        <kml:Point>
            <gx:altitudeMode>relativeToSeaFloor</gx:altitudeMode>
            <kml:coordinates>5.550886412795313,45.06638495609347,1008</kml:coordinates>
        </kml:Point>
    </met:lieu>
    <!-- Bulletins d'exemple -->
    <met:bulletin date="2017-12-16T06:00:00+01:00">
```

```
        <met:température>-6.5</met:température>
        <met:anémométrie>
            <met:vitesse>10</met:vitesse>
        </met:anémométrie>
        <met:précipitations type="neige">
            <met:niveau>15.0</met:niveau>
        </met:précipitations>
    </met:bulletin>
    <met:bulletin date="2017-12-16T12:00:00+01:00">
        <met:température isothermeZero="600">-2.0</met:température>
        <met:anémométrie direction="NE">
            <met:vitesse>30</met:vitesse>
            <met:rafales>55</met:rafales>
        </met:anémométrie>
        <met:précipitations type="aucune"/>
    </met:bulletin>
    <!--
        A COMPLETER : Bulletins du 21/12/17
    -->
</met:météo>
```

On donne également le squelette du schéma XML correspondant:
→ Fichier ***meteo.xsd*** :

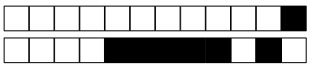
```
<?xml version="1.0"?>
<xs:schema version="1.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:met="http://www.meteo-france.fr/bulletin"
           xmlns:kml="http://www.opengis.net/kml/2.2"
           targetNamespace="http://www.meteo-france.fr/bulletin"
           elementFormDefault="qualified">

    <xs:import namespace="http://www.opengis.net/kml/2.2"
               schemaLocation="http://schemas.opengis.net/kml/2.2.0/ogckml22.xsd"/>
    <xs:import namespace="http://www.google.com/kml/ext/2.2"
               schemaLocation="http://code.google.com/apis/kml/schema/kml22gx.xsd"/>

    <!-- élément racine -->
    <!-- Type Météo -->
    <!-- Type Période -->
    <!-- Type Bulletin -->
    <!-- Type ValeurTempérature -->
    <!-- Type Température -->
    <!-- Type Anémométrie -->
    <!-- Type Direction -->
    <!-- Type Précipitations -->
    <!-- Type TypePrécipitation -->
    <!-- Type DoublePositif -->

</xs:schema>
```

L'instance de document XML utilise les types `Précipitations`, `Température`, `Anémométrie`, `Bulletin` et `Météo`. Ces types se réfèrent à d'autres types : les types `TypePrécipitation`, `ValeurTempérature` et `Direction` représentent respectivement les différents types de précipitations, les valeurs que peuvent prendre la température et les différentes directions du vent. Un type `DoublePositif` est utilisé pour les valeurs de vitesse du vent et les niveaux de précipitations (positives). Enfin un type `Période` a pour attributs `année` et `mois` correspondants aux types XML Schéma standard `xs:gYear` et `xs:gMonth`. Les renseignements donnés dans le texte introductif donnent leur description complète.
NB: Le type XML Schéma standard de date employé, `xs:dateTime`, encode la date, l'heure et le fuseau horaire ; la date et l'heure sont séparés par la lettre 'T'. Le lieu est de type `kml:PlacemarkType` du langage `KML`.



Question 1 ♣ Les types (plusieurs cases peuvent être cochées) :

- ☐ Le type `Période` est un type simple
- ☐ L'élément `précipitations` peut être vide
- ☐ La racine du document XML est de type `Bulletin`
- ☐ Un `Bulletin` comprend au moins un élément de type `Précipitations`, un élément de type `Température` et un élément de type `Anémométrie`
- ☐ Un `DoublePositif` est un type simple avec une restriction de type "pattern"
- ☐ `TypePrécipitations`, `Direction` et `ValeurTempérature` sont des types simples
- ☐ Le type `Direction` est un type simple avec une restriction de type "pattern"
- ☐ *Aucune de ces réponses n'est correcte.*

Question 2 Dans l'espace réservé ci-dessous, modélisez les types `TypePrécipitation`, `ValeurTempérature` et `Direction`. Ecrivez également le type `DoublePositif`. Modélisez enfin le type `Période`.

☐**A** ☐**B** ☐**C** ☐**D** ☐**E** *Réservé au correcteur*

Question 3 Dans l'espace réservé ci-dessous, modélisez les types `Précipitations`, `Température` et `Anémométrie` de telle manière que cela corresponde à l'instance de document et aux renseignements donnés dans le texte. Ecrivez également le type `Bulletin` et le type `Météo`. Ecrivez enfin l'élément racine.

☐**A** ☐**B** ☐**C** ☐**D** ☐**E** *Réservé au correcteur*

Question 4 Il est précisément 11h du matin. Dans l'espace réservé ci-dessous, complétez l'instance de document XML avec les informations dont vous disposeriez pour la météo de la journée d'aujourd'hui (le 21/12/17) à cette heure de la journée. On supposera un beau temps, avec des températures comprises entre 0 et 3°C, sans vent le matin, pouvant se lever jusqu'à 40 km/h l'après midi, toute la journée. ☐**A** ☐**B** ☐**C** ☐**D** ☐**E** *Réservé au correcteur*

2 Analyse DOM et affichage Java du document RSS(5 points)

Une application Java est proposée pour lire les flux RSS. On s'intéresse ici à l'analyse du document par un parseur DOM et à son affichage. Le code Java suivant est fourni. → Fichier ***MeteoRSS.java*** :

```
1  public class MeteoRSS {
2
3      // Analyse un document XML et retourne un Document DOM
4      private static Document parseXML(String path, String filename) throws
           ParserConfigurationException, SAXException, IOException{
5          DocumentBuilderFactory domFactory = DocumentBuilderFactory.newInstance();
6          domFactory.setNamespaceAware(true);
7          DocumentBuilder domBuilder = domFactory.newDocumentBuilder();
8          return domBuilder.parse(new File(path + "/" + filename));
9      }
10
11     // Récupère la date du jour au format yyyy-mm-dd
12     private static String getRSSDate() {
13         GregorianCalendar gregorianCalendar=new GregorianCalendar(); // récupère la date
               courante
14         String date = String.valueOf(gregorianCalendar.get(GregorianCalendar.YEAR)); // récupé
               re l'année
15         date +="-"+(gregorianCalendar.get(GregorianCalendar.MONTH)+1); // récupére le mois
16         date +="-"+gregorianCalendar.get(GregorianCalendar.DAY_OF_MONTH); // récupére le jour
17         return date;
18     }
19 }
```

Question 5 Ajoutez un attribut de classe permettant de stocker un document DOM dans l'instance de classe, ainsi qu'un constructeur prenant un chemin de répertoire et un nom de fichier comme paramètre. Complétez également la méthode `printRSS():void` pour qu'elle affiche ce qui est demandé dans les commentaires. NB: A toutes fins utiles, on rappelle l'existance des méthodes `Element.getElementsByTagName(name:String):NodeList`, `NodeList.getLength():int`, `NodeList.item(index:int):Node`, `Node.getChildNodes():NodeList`, et `Node.getTextContent():String`.

```
1  public class MeteoRSS {
2
3      // Attribut
4
5      // Constructeur
6
7
8
9
10     // [Fourni] Analyse un document XML et retourne un Document DOM
11     private static Document parseXML(String path, String filename) // ...
12     // [Fourni] Récupère la date du jour au format yyyy-mm-dd
13     private static String getRSSDate() // ...
14
15
16     public void printRSS(){
17         try {
18             // On récupère l'élement racine à partir de l'attribut _rss:DOM
19             Element rootElt = _rss.getDocumentElement(); // fourni
20             // on récupére l'élément "channel"
21             Element channelElt =
22             // On affiche la description du channel
23
24
25
26
27
28             // on récupére la liste des noeuds "item"
29             NodeList itemNL =
30             // ici on affiche le titre, la date et heure de publication
31             // et la description de chacun des items du flux RSS
32
33
34
35
36
37
38
39
40
41
42
43         } catch (ParserConfigurationException | SAXException | IOException
               ex) {
44             Logger.getLogger(Meteo.class.getName()).log(Level.SEVERE, null,
                   ex);
45         }
46     }
47 }
```

..... ☐A ☐B ☐C ☐D ☐E *Réservé au correcteur*