Eindtoets XML: Theorie en toepassingen

Deze eindtoets geeft een indruk, niet meer en niet minder, van mogelijke tentamenvragen. De spreiding over verschillende onderwerpen zal nooit bij elk tentamen hetzelfde zijn. Diverse onderwerpen komen in deze eindtoets niet aan bod, maar dat betekent niet dat er in de echte tentamens ook niets over zal worden gevraagd.

Alle opgaven hebben betrekking op het volgende XML-document museum.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<museum>
  <zalen>
    <zaal zaalnr="z101">Blauwe zaal</zaal>
    <zaal zaalnr="z102">Groene zaal</zaal>
    <zaal zaalnr="z201">Rode zaal</zaal>
  </zalen>
  <objecten>
    <object type="schilderij" zaalnr="z101">
      <kunstenaar>Rembrandt</kunstenaar >
      <titel>Nachtwacht</titel>
    </object>
    <object type="beeld" zaalnr="z101">
      <kunstenaar>Michelangelo</kunstenaar>
      <titel>David</titel>
    </object>
    <object type="boek" zaalnr="z201">
      <auteur>Multatuli</auteur>
      <titel>Max Havelaar</titel>
      <uitgever>Schadd</uitgever>
      <jaar>1871</jaar>
    </object>
    <object type="boek" zaalnr="z201">
      <auteur>Betje Wolff</auteur>
      <auteur>Aagje Deken</auteur>
      <titel>Sara Burgerhart</titel>
      <jaar>1782</jaar>
    </object>
  </objecten>
</museum>
```


-

Opgave 1 (DTD - 10 punten)

Gegeven is dat museum.xml valide is met betrekking tot de volgende onvolledige DTD museum.dtd. Er zijn twee genummerde stukjes uit de DTD verwijderd.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT museum     (zalen, objecten)>
<!ELEMENT zalen      (zaal+)>
<!ELEMENT zaal       (#PCDATA)>
<!ELEMENT objecten   (object)+>
<!ELEMENT object     [..2..]>
<!ELEMENT kunstenaar (#PCDATA)>
<!ELEMENT titel      (#PCDATA)>
<!ELEMENT auteur     (#PCDATA)>
<!ELEMENT uitgever   (#PCDATA)>
<!ELEMENT jaar       (#PCDATA)>
<!ATTLIST zaal
          zaalnr ID #REQUIRED>
<!ATTLIST object
          [..1..]
          zaalnr IDREF #REQUIRED>
```

Vul de ontbrekende stukken aan volgens de gegeven omschrijving.

`[..1..]` het verplichte attribuut `type`, waarbij uit de volgende lijst van mogelijkheden moet worden gekozen: schilderij, beeld, boek.

`[..2..]` een content model voor het element object, ervan uitgaande dat een object ofwel één kunstenaar en één titel heeft, ofwel één of meer auteurs, één titel, geen of één uitgever en één jaar. De volgorde is hierbij van belang.

-

Opgave 2 (XML-Schema - 12 punten)

Hieronder ziet u een onvolledig schema voor museum.xml. Er zijn twee stukken uit het schema verwijderd en vervangen door genummerde tekst. Geef de ontbrekende stukken code (en niet meer dan dat).

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="museum" type="museumType"/>

  <xsd:complexType name="museumType">
  [..1.. Een element van type museumType bevat één element zalen van type zalenType
         en één element objecten van type objectenType, in willekeurige volgorde
  ]
  </xsd:complexType>

  <xsd:complexType name="zalenType">
    <xsd:sequence>
      <xsd:element name="zaal" type="zaalType" minOccurs="1"
                                               maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="objectenType">
    <xsd:sequence>
      <xsd:element name="object" type="objectType" minOccurs="1"
                                                   maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="zaalType">
     <xsd:simpleContent>
       <xsd:extension base="xsd:string">
         <xsd:attribute name="nr" type="xsd:string"/>
       </xsd:extension>
     </xsd:simpleContent>
  </xsd:complexType>

  <xsd:complexType name="objectType">
  [..2.. Een element van type objectType bevat óf een groep beeldendeKunst-Group of
         een groep boek-Group. Verder bevat het een string-attribuut type en een
         string-attribuut zaalnr ]
  </xsd:complexType>

  <xsd:group name="beeldendeKunst-Group">
    <xsd:sequence>
      <xsd:element name="kunstenaar" type="xsd:string" minOccurs="1"
                                                       maxOccurs="1"/>
      <xsd:element name="titel" type="xsd:string" minOccurs="1" maxOccurs="1"/>
    </xsd:sequence>
  </xsd:group>

 <xsd:group name="boek-Group">
   <xsd:sequence>
     <xsd:element name="auteur" type="xsd:string" minOccurs="1"
                                                  maxOccurs="unbounded"/>
     <xsd:element name="titel" type="xsd:string" minOccurs="1" maxOccurs="1"/>
     <xsd:element name="uitgever" type="xsd:string" minOccurs="0" maxOccurs="1"/>
     <xsd:element name="jaar" type="xsd:decimal" minOccurs="1" maxOccurs="1"/>
   </xsd:sequence>
  </xsd:group>

</xsd:schema>
```


-

Opgave 3 (XPath – 6 punten)

a Geef precies aan welke nodeset van museum.xml wordt geselecteerd door de volgende XPath-expressie: `//object[2]/following-sibling::*/auteur`

b Schrijf de expressie van onderdeel a in volledig geëxpandeerde vorm.

c Schrijf een XPath-expressie die het aantal objecten uit zaal z101 selecteert.

Opgave 4 (XSLT – 6 punten)

De volgende, onvolledige XSLT-stylesheet moet, in de vorm van een HTML-tabel, een overzicht opleveren van alle schilderijen met hun kunstenaar, titel en zaal.

Welk code moet in de plaats komen van het commentaar?

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
<xsl:template match="/museum">
<html><body>

<h3>Overzicht schilderijencollectie</h3>
<p />

<table border="1" cellpadding="5">
   <tr>
      <th>kunstenaar</th>
      <th>titel</th>
      <th>zaal</th>
   </tr>

   <!—- vervang dit commentaar door de juiste code -->

</table>

</body></html>
</xsl:template>
</xsl:stylesheet>
```


-

Opgave 5 (Namespaces/target namespace - 8 punten)
We voorzien het museum-document van namespaces, als volgt:

```
<?xml version="1.0" encoding="UTF-8"?>
<museum xmlns="http://www.rijksmuseum.nl/gebouw"
        xmlns:mc="http://www.rijksmuseum.nl/collectie”>
    <zalen>
        <zaal zaalnr="z101">Blauwe zaal</zaal>
        <zaal zaalnr="z102">Groene zaal</zaal>
        <zaal zaalnr="z201">Rode zaal</zaal>
    </zalen>
    <mc:objecten>
        <object>
        ...
        </object>
        ...
        ...
    </mc:objecten>
</museum>
```

Geef voor elk der namen museum, zalen, zaal, zaalnr en object aan welke mogelijkheid (A, B of C) van toepassing is:
A de naam behoort tot de namespace `http://www.rijksmuseum.nl/gebouw`
B de naam behoort tot de namespace `http://www.rijksmuseum.nl/collectie`
C de naam behoort niet tot een namespace.

Geef uw antwoord in de vorm van een lijstje met de namen en achter elke naam de juiste letter, zonder toelichting.

-

Opgave 6 (Stax – 15 punten)

We maken een Javaprogramma KunstZoeker dat de namen van alle zalen afdrukt waarin een object van kunstenaar Rembrandt zich bevindt. We willen in het programma gebruik maken van XMLStreamReader.
Globaal ziet de code er aldus uit:

Lees de informatie over zalen, maak Zaal-objecten en voeg ze toe aan ArrayList<Zaal> zalen.
Lees de informatie over objecten, zoek daarin naar kunstenaar Rembrandt en leg het betreffende zaalnr vast. Zoek op basis van dit zaalnr het overeenkomende Zaal-object en druk daarvan de naam van de zaal af. Herhaal dit voor ieder object met kunstenaar Rembrandt. Voor het gemak van deze opgave beginnen we na het lezen van de zalen weer aan het begin van het document om de kunstobjecten te lezen.

```
public class KunstZoeker {

  public static final String FILENAAM       = "museum.xml";
  public static final String ZOEKKUNSTENAAR = "Rembrandt";
  public static final String KUNSTENAAR     = "kunstenaar";
  public static final String OBJECT         = "object";
  public static final String ZAALNR         = "zaalnr";
  public static final String ZAAL           = "zaal";

  private ArrayList<Zaal>    zalen          = new ArrayList<Zaal>();

  public static void main(String[] args) throws Exception {
    KunstZoeker k = new KunstZoeker();
    XMLInputFactory inpf = XMLInputFactory.newInstance();
    InputStream is = new FileInputStream(new File(FILENAAM));
    XMLStreamReader xr = inpf.createXMLStreamReader(is);
    String zaalnr = "";
    String zaalnaam = "";
    // Lees Zalen uit XML-document , maak Zaal-objecten en voeg toe aan zalen
    [[..1..]]
    is = new FileInputStream(new File(FILENAAM));
    xr = inpf.createXMLStreamReader(is); // begin weer opnieuw
    boolean kunstenaargevonden = false;
    // Lees Objecten en druk zaalnamen af van kunstenaar Rembrandt
    [[..2..]]
  }

  public Zaal getZaal(String nr) {
    Zaal z = null;
    for (Zaal zz : zalen) {
      if (zz.getZaalNr().equals(nr)) {
        z = zz;
      }
    }
    return z;
  }

  public void addZaal(String zaalnr, String zaalnaam) {
    Zaal z = new Zaal(zaalnr, zaalnaam);
    zalen.add(z);
  }
}
```

en

```
public class Zaal {
  private String zaalNr = "";
  private String zaalNaam = "";

  public Zaal(String zaalNr, String zaalNaam) {
    super();
    this.zaalNr = zaalNr;
```

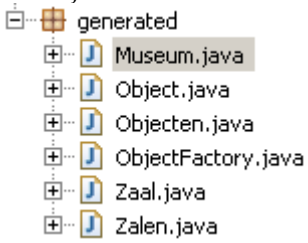

-

```
    this.zaalNaam = zaalNaam;
  }

  public String getZaalNr() {
    return zaalNr;
  }

  public String getZaalNaam() {
    return zaalNaam;
  }
}
```

a Geef de code voor onderdeel 1
b Geef de code voor onderdeel 2

Opgave 7 (Jaxb – 15 punten)

We gaan hetzelfde probleem oplossen als van opgave 6 maar gebruiken nu JAXB. museum.xml heeft een bijbehorend schema museum.xsd. Op basis van dit schema laten we XJC klassen genereren.



De relevante benodigde klassen hebben de volgende code (we hebben allerlei niet relevante zaken weggelaten)

```
public class Museum {
    protected Zalen zalen;
    protected Objecten objecten;

    public Zalen getZalen() {
        return zalen;
    }

    public Objecten getObjecten() {
        return objecten;
    }
}

public class Zalen {
    protected List<Zaal> zaal;

    public List<Zaal> getZaal() {
        if (zaal == null) {
            zaal = new ArrayList<Zaal>();
        }
        return this.zaal;
    }
}

public class Zaal {
    protected String content;
    protected String zaalnr;

    public String getContent() {
        return content;
    }
    public String getZaalnr() {
        return zaalnr;
    }
}
```


-

```
public class Object {
    protected String kunstenaar;
    protected String type;
    protected String zaalnr;

    public String getKunstenaar() {
        return kunstenaar;
    }
    public String getType() {
        return type;
    }
    public String getZaalnr() {
        return zaalnr;
    }
}
```

We maken een klasse JaxbKunstZoeker met de volgende code:

```
public class JaxbKunstZoeker {

  public static final String FILENAAM       = "museum.xml";
  public static final String ZOEKKUNSTENAAR = "Rembrandt";
  public static final String TYPE           = "schilderij";

  public static void main(String[] args) throws Exception{
    JAXBContext context = JAXBContext.newInstance("generated");
    Unmarshaller um = context.createUnmarshaller();
    File file = new File(FILENAAM);
    // Lees bestand en maak List<Zaal> zalenlijst
   [[..1..]]
    // Maak List<Object> en doorloop deze. Als type = “schilderij” en
    // kunstenaar = “Rembrandt”, zoek zaalnaam op en druk af
   [[ ..2..]]
  }

  public static String getZaalNaam(List<Zaal> lijst,String zaalnr){
    String zaalnaam = "";
    for (Zaal z:lijst){
      if(z.getZaalnr().equals(zaalnr)){
        zaalnaam = z.getContent();
      }
    }
    return zaalnaam;
  }
}
```

a Geef de code voor onderdeel 1
b Geef de code voor onderdeel 2

c Zowel de code van opgave 6 als de code van deze opgave is vrij omvangrijk. Als laatste maken we een programma op basis van een XQuery-expressie die opgeslagen is in een document q1.xquery. Het resultaat van de XQuery-expressie met museum.xml als uitgangsdocument luidt:

```
<?xml version="1.0" encoding="UTF-8"?>
<zalen>
   <zaal>Blauwe zaal</zaal>
</zalen>
```

We maken met XOM een programma XOMKunstZoeker dat gebruik maakt van q1.xquery. We willen als uitvoer : “Rembrandt gevonden in Blauwe zaal”.

Gegeven de volgende code

```
public class XOMKunstZoeker {
  public static final String QUERYNAAM = "q1.xquery";
  public static final String TEKST     = "Rembrandt gevonden in ";

  public static void main(String[] args) throws Exception {
```


-

```
    [[..1..]]
  }

  public static String getXQuery(String filenaam) throws Exception {
    BufferedReader br = new BufferedReader(new FileReader(new File(QUERYNAAM)));
    StringBuffer sb = new StringBuffer();
    String s = null;
    while ((s = br.readLine()) != null) {
      sb.append(s);
    }
    return new String(sb);
  }
}
```

Geef de benodigde code voor [[..1..]]

Opgave 8 (XQuery – 12 punten)
We willen op basis van museum.xml een nieuw XML-document maken. In het nieuwe document zijn de objecten gegroepeerd naar type, waarbij de verschillende typen alfabetische zijn gesorteerd. Uitgaande van museum.xml zijn de mogelijke typen: beeld, boek en schilderij. De resulterende XML-file op basis van museum.xml ziet er als volgt uit:

```
<?xml version="1.0" encoding="UTF-8"?>
<objectenPerType>
  <beeld>
    <titel>David</titel>
  </beeld>
  <boek>
    <titel>Max Havelaar</titel>
    <titel>Sara Burgerhart</titel>
  </boek>
  <schilderij>
    <titel>Nachtwacht</titel>
  </schilderij>
</objectenPerType>
```

Opdracht
Vervang het grijs gemarkeerde deel in onderstaande query door een expressie, zodat de gevraagde output wordt gegenereerd.

```
element objectenPerType {
   [ -- ontbrekende code -- ]
}
```


-

Opgave 9 (WSDL - 8)

Gegeven is het onderstaande WSDL-document:

```
<wsdl:definitions xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/
  xmlns:ns1=http://org.apache.axis2/xsd
  xmlns:wsaw=http://www.w3.org/2006/05/addressing/wsdl
  xmlns:http=http://schemas.xmlsoap.org/wsdl/http/
  xmlns:ns0=http://ws.apache.org/axis2
  xmlns:xs=http://www.w3.org/2001/XMLSchema
  xmlns:mime=http://schemas.xmlsoap.org/wsdl/mime/
  xmlns:soap=http://schemas.xmlsoap.org/wsdl/soap/
  xmlns:soap12=http://schemas.xmlsoap.org/wsdl/soap12/
  targetNamespace="http://ws.apache.org/axis2">

  <wsdl:types>
    <xs:schema xmlns:ns=http://ws.apache.org/axis2
       attributeFormDefault="qualified"
       elementFormDefault="qualified"
       targetNamespace="http://ws.apache.org/axis2">
      <xs:element name="getBoekenResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="0" name="return"
                        nillable="true" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>

  <wsdl:message name="getBoekenRequest"/>
  <wsdl:message name="getBoekenResponse">
    <wsdl:part name="parameters" element="ns0:getBoekenResponse"/>
  </wsdl:message>

  <wsdl:portType name="MuseumServicePortType">
    <wsdl:operation name="getBoeken">
      <wsdl:input message="ns0:getBoekenRequest" wsaw:Action="urn:getBoeken"/>
      <wsdl:output message="ns0:getBoekenResponse"
                   wsaw:Action="urn:getBoekenResponse"/>
    </wsdl:operation>
  </wsdl:portType>

  <wsdl:service name="MuseumService">
    <wsdl:port name="MuseumServiceSOAP11port_http"
               binding="ns0:MuseumServiceSOAP11Binding">
      <soap:address location="http://localhost:8080/axis2/services/MuseumService"/>
    </wsdl:port>
    <wsdl:port name="MuseumServiceSOAP12port_http"
               binding="ns0:MuseumServiceSOAP12Binding">
      <soap12:address
                location="http://localhost:8080/axis2/services/MuseumService"/>
    </wsdl:port>
    <wsdl:port name="MuseumServiceHttpport" binding="ns0:MuseumServiceHttpBinding">
      <http:address location="http://localhost:8080/axis2/services/MuseumService"/>
    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```

Vragen

a Geef een gedetailleerde functionele beschrijving van de service(s) die volgens dit WSDL document wordt (worden) beschreven, dat wil zeggen:
-- Welke methode(n) wordt(worden) geboden?
-- Welke input en output parameters horen daarbij en wat is het type van deze parameters?

b Motiveer waarom het element `<wsdl:message name="getBoekenRequest"/>` een leegelemnent is.

c In hoeverre is dit WSDL document compleet? Als het incompleet is, welke onderdelen worden gemist?

-

Opgave 10 (SOAP- 8 punten)

Geef aan welke fouten voorkomen in het volgende SOAP request- respectievelijk response-bericht bij de aanroep van getBoeken. Let daarbij op het WSDL document uit de vorige opgave. Motveer uw antwoorden.

a SOAP request-bericht:

```
POST /axis2/services/MuseumService HTTP/1.1
Content-Type: application/soap+xml; charset=UTF-8; action="urn:getBoeken"
User-Agent: Axis2
Host: 127.0.0.1:8082
Transfer-Encoding: chunked

<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
  <soapenv:Body>
    <ns1:getBoeken xmlns:ns1="http://ws.apache.org/axis2">
      <ns1:nummer>101</ns1:nummer>
    </ns1:getBoeken>
  </soapenv:Body>
</soapenv:Envelope>
```

b SOAP response-bericht:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/soap+xml; action="urn:getBoekenResponse";charset=UTF-8
Transfer-Encoding: chunked
Date: Thu, 18 Mar 2010 13:07:56 GMT

<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv=”http://www.w3.org/2003/05/soap-envelope”
                  xmlns:xsd=”http://www.w3.org/2001/XMLSchema”
                  xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ns:getBoekenResponse xmlns:ns="http://ws.apache.org/axis2">
      <ns:return xsi:type="xsd:token>Max Havelaar</ns:return>
      <ns:return xsi:type="xsd:token >Sara Burgerhart</ns:return>
    </ns:getBoekenResponse>
  </soapenv:Body>
</soapenv:Envelope>
```


-

Uitwerking van de opgaven

Opgave 1 (10 punten)

```
<?xml version="1.0" encoding="UTF-8"?>
<!--elementen en attributen-->
<!ELEMENT museum     (zalen, objecten)>
<!ELEMENT zalen      (zaal+)>
<!ELEMENT zaal       (#PCDATA)>
<!ELEMENT kunstenaar (#PCDATA)>
<!ELEMENT titel      (#PCDATA)>
<!ELEMENT objecten   (object)+>
<!ELEMENT object     ((kunstenaar, titel) | (auteur+, titel, uitgever?, jaar))>
<!ELEMENT auteur     (#PCDATA)>
<!ELEMENT uitgever   (#PCDATA)>
<!ELEMENT jaar       (#PCDATA)>
<!ATTLIST object
   type (schilderij | beeld | boek) #REQUIRED
   zaalnr IDREF #REQUIRED)>
<!ATTLIST zaal
          zaalnr ID #REQUIRED>
```

Opgave 2 (12 punten)

Fragment 1:

```
<xsd:all>
  <xsd:element name="zalen" type="zalenType" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="objecten" type="objectenType" minOccurs="1" maxOccurs="1"/>
</xsd:all>
```

Omdat binnen een xsd:all-element minOccurs en maxOccurs defaultwaarden 1 hebben, mogen de minOcurs- en maxOccurs-aanduidingen ook worden weggelaten.

Fragment 2:

```
<xsd:choice minOccurs="1" maxOccurs="1">
  <xsd:group ref="beeldendeKunst-Group"/>
  <xsd:group ref="boek-Group"/>
</xsd:choice>

<xsd:attribute name="type" type="xsd:string"/>
<xsd:attribute name="zaalnr" type="xsd:string"/>
```

Opgave 3 (6 punten)

a Geselecteerde node set: de auteur-nodes horende bij Multatuli, Betje Wolff, Aagje Deken

b In geëxpandeerde vorm:
`/descendant-or-self::node()/child::object[2.0]/following-sibling::*/child::auteur`

c Maximaal verkorte vorm: `count(//object[@zaalnr="z101"])`

Bij c zijn meer of minder geëxpandeerde vormen natuurlijk ook goed. De maximaal geëxpandeerde vorm is: `count(/descendant-or-self::node()/child::object[(attribute::zaalnr="z101")])`
Verder mag `//object` worden vervangen door onder meer: `/museum/objecten/object` (of een geëxpandeerde vorm daarvan).

-

Opgave 4 (6 punten)

We tonen twee mogelijke oplossingen: één zonder en één met gebruik van templates.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
<xsl:template match="/museum">
<html><body>

<h3>Overzicht schilderijencollectie</h3>
<p />

<table border="1" cellpadding="5">
   <tr>
      <th>kunstenaar</th>
      <th>titel</th>
      <th>zaal</th>
   </tr>

   <xsl:for-each select="objecten/object" >
      <xsl:if test="@type='schilderij'" >
         <tr>
            <td> <xsl:value-of select="kunstenaar" /> </td>
            <td> <xsl:value-of select="titel" /> </td>
            <td> <xsl:value-of select="@zaal" /> </td>
         </tr>
      </xsl:if>
   </xsl:for-each>

</table>

</body></html>
</xsl:template>
</xsl:stylesheet>
```

Voor de XPath-expressies zijn verschillende varianten mogelijk, bijvoorbeeld:
– `@` mag worden vervangen door `attribute::`
– `objecten/object` mag worden vervangen door `//object`.

Verder kan de test ook worden opgenomen in de XPath-expressie. De gevraagde code wordt dan:

```
   <xsl:for-each select="objecten/object[@type='schilderij']" >
         <tr>
            <td> <xsl:value-of select="kunstenaar" /> </td>
            <td> <xsl:value-of select="titel" /> </td>
            <td> <xsl:value-of select="@zaal" /> </td>
         </tr>
   </xsl:for-each>
```


-

Een alternatief met gebruik van templates is als volgt:

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                version="1.0">
<xsl:template match="/museum">
<html><body>

<h3>Overzicht schilderijencollectie</h3>
<p />

<table border="1" cellpadding="5">
   <tr>
      <th>kunstenaar</th>
      <th>titel</th>
      <th>zaal</th>
   </tr>
   <xsl:apply-templates select="objecten/object[@type='schilderij']"/>
</table>

</body></html>
</xsl:template>

<xsl:template match="object[@type='schilderij']">
   <tr>
      <td> <xsl:value-of select="kunstenaar" /> </td>
      <td> <xsl:value-of select="titel" /> </td>
      <td> <xsl:value-of select="@zaal" /> </td>
   </tr>
</xsl:template>

</xsl:stylesheet>
```

Opgave 5 (8 punten)

Wanneer attribuutnamen niet expliciet zijn gekwalificeerd (colonized), horen ze niet tot een namespace (unqualified). Daarom hoort zaalnr niet tot een namespace (C). Alle overige namen horen tot de default namespace (A).

Opmerkingen

1 Ook object hoort tot de default namespace; elementen erven immers *niet* de namespace van hun ouderelement.

2 De namespace-alias mc wordt gedefinieerd via een attribuut van het element museum. Dat houdt echter geen koppeling in aan de bij mc horende namespace. De naam 'museum' valt daarom onder de default namespace.

Opgave 6 (15 punten)

a

```
// Lees Zalen
while (xr.hasNext()) {
  if ((xr.next() == XMLStreamConstants.START_ELEMENT)
       && xr.getLocalName().equals(ZAAL)) {
    zaalnr = xr.getAttributeValue(null, ZAALNR);
    zaalnaam = xr.getElementText();
    k.addZaal(zaalnr, zaalnaam);
  }
}
```


-

```
b
boolean kunstenaargevonden = false;
// Lees Objecten en druk zalen af
while (xr.hasNext()) {
  if ((xr.next() == XMLStreamConstants.START_ELEMENT)
    && xr.getLocalName().equals(OBJECT)) {
    zaalnr = xr.getAttributeValue(null, ZAALNR);
    while (xr.hasNext() && !kunstenaargevonden) {
      kunstenaargevonden = ((xr.next() == XMLStreamConstants.START_ELEMENT)
      && xr.getLocalName().equals(KUNSTENAAR) && xr.getElementText()
      .equals(ZOEKKUNSTENAAR));
      }
      if (kunstenaargevonden) {
        zaalnaam = k.getZaal(zaalnr).getZaalNaam();
        System.out.println("gevonden en hangt in " + zaalnaam);
      }
      kunstenaargevonden = false;
   }
}
```

Opgave 7 (15 punten)

a

```
Museum museum = (Museum) um.unmarshal(file);
Zalen zalen = museum.getZalen();
List<Zaal> zalenlijst = zalen.getZaal();
```

```
b

Objecten objecten = museum.getObjecten();
List<Object> objectenlijst = objecten.getObject();
for (Object ob:objectenlijst){
  if(ob.getType().equals(TYPE) && ob.getKunstenaar().equals(ZOEKKUNSTENAAR)){
    String zaalnr = ob.getZaalnr();
    System.out.println(getZaalNaam(zalenlijst,zaalnr));
  }
}
```

c

```
public static void main(String[] args) throws Exception {
    Nodes nodes = XQueryUtil.xquery(null, getXQuery(QUERYNAAM));
    for (int i = 0; i < nodes.size(); i++) {
      Node n = nodes.get(i);
      System.out.println(TEKST + n.getValue());
    }
  }
```

Opgave 8 (12 punten)

```
element objectenPerType {
   for $t in distinct-values(doc("museum.xml")//object/@type)
   order by $t ascending
   return element {$t} {

       for $o in doc("museum.xml")//object
       where $o/@type = $t
       return $o/titel
   }
}
```

Opgave 9 (8 punten)

a Er wordt één service geboden, namelijk: `String[] getBoeken();` Dus een methode getBoeken, die geen input parameters heeft en een output parameter (een array van strings of iets dergelijks) .

b Methode getBoeken heeft geen inputparameters. Daarom is het message element leeg.

c Dit WSDL document is incompleet. Gemist worden de bindingelementen voor SOAP en HTTP.

-

Opgave 10 (8 punten)

a In het request bericht wordt een inputparameter meegegeven van het type int. Deze is er volgens het WSDL document niet.

b In het response-bericht is het terugkeertype een array van tokens, terwijl dat volgens het WSDL document een array van strings moet zijn.

-