Podstawy XML i XML Schema

Celem ćwiczenia jest zapoznanie z dokumentami XML, XML Schema oraz ich walidacją.
Do wykonania ćwiczenia potrzebny jest dowolny edytor plików tekstowych, przeglądarka internetowa oraz walidator.

Do walidowania plików należy użyć walidatora http://tools.decisionsoft.com/schemaValidate/.

W zadaniu 2 następuje sprawdzanie czy plik XML jest **poprawnie uformowany** (*ang. well-formed*) tzn. zgodny z zasadami tworzenia dokumentów XML. W pozostałych zadaniach sprawdzamy dodatkowo, czy plik XML jest **poprawny strukturalnie** (*ang. valid document*), tzn. czy zawiera elementy, atrybuty, hierarchię zgodne z gramatyką zawartą w pliku XML Schema.

1. Na dysku wskazanym przez prowadzącego stworzyć katalog nazwany własnym imieniem i nazwiskiem. Umieścić w nim pliki ściągnięte z Moodla. Do pracy z plikami można użyć środowiska Visual lub zwykłego notatnika. Po zajęciach własny katalog należy **SKASOWAĆ**. Należy pamiętać o okresowym zachowywaniu wyników pracy. Po każdym punkcie plik należy **walidować**, aby sprawdzić czy jest zgodny z tworzonym XML Schema.
2. (0,5pkt) W pliku HH0.xml popełniono kilka błędów. Znajdź je i popraw tak, aby plik miał poprawną składnię i parser nie zgłaszał błędów.
3. **Poprosić prowadzącego o sprawdzenie pracy**
4. Zapoznać się z plikami udostępnionymi na Moodlu. Przeanalizować plik HH.xml, zwrócić uwagę na strukturę dokumentu, wykorzystane znaczniki. Przeanalizować plik HH.xsd, zwrócić uwagę na sposób definiowania znaczników występujących w pliku HH.xml. Wyświetlić pliki xml i xsd w przeglądarce. Zastanowić się, dlaczego są one wyświetlane w taki sposób. Dane potrzebne do uzupełniania pliku XML znajdują się w pliku text.docx – należy dodawać je zgodnie z poleceniami zawartymi w instrukcji.
5. (1pkt) W pliku XML dodać swoje imię i nazwisko. W pliku xsd dodać odpowiednią deklarację. Wykorzystać zadeklarowany element `author` i referencje.

```
<course>
   <author>
      <name>student's name</name>
      <surname>student's surname</surname>
   </author>
```

6. **Poprosić prowadzącego o sprawdzenie pracy**
7. (0,5pkt) Do elementu `activities` dodać podelement `score`, w którym przechowywana będzie punktacja. Dodać odpowiednią deklarację w xsd. Podelement `score` powinien zostać zadeklarowany jako ostatni podelement elementu `activities`.
8. (0,5pkt) W pliku XML do elementu `activities` dodać atrybut `id` typu `byte`. Odpowiednio zmienić definicję typu `activitiesType`. Atrybut opisuje kolejny numer aktywności.

```
<activities id="1">
   <topic>Hypertext and hypermedia</topic>
   <range>
      <component>Hypertext &amp; hypermedia</component>
      <component>HTML CSS</component>
      <component>XML</component>
      <component>XML Schema</component>
      <component>DTD</component>
      <component>XSLT</component>
      <component>FO</component>
   </range>
   <score>30</score>
```

9. **Poprosić prowadzącego o sprawdzenie pracy**

10. (1pkt) W pliku XML umieścić linki. Zadeklarować element `links`, w którym można umieścić dowolną liczbę elementów `link`. W elemencie `link` atrybut `source` ma zawierać adres, natomiast tekst linku ma być umieszczony jako wartość elementu `link`. W pliku xsd dodać odpowiednie definicje. Element `links` ma być zadeklarowany jako opcjonalny – może nie wystąpić w pliku XML. Element `links` ma być podelementem elementu `information`. Typ dla elementu `link` zdefiniować globalnie.

```
<links>
   <link source="https://www.w3schools.com/html/">HTML w3schools</link>
   <link source="https://www.w3schools.com/xml/default.asp">XML w3schools</link>
   <link source="https://enauczanie.pg.edu.pl/moodle/">Moodle</link>
</links>
```

11. **Poprosić prowadzącego o sprawdzenie pracy**
12. (0,5pkt) Umieścić w pliku XML element pozwalający na przechowywanie adresu plików zdjęć oraz ich tytułu. W pliku xsd dodać odpowiednią definicję. Wykorzystać zdefiniowany w poprzednim punkcie typ.

```
<media>
   <image source="img/eti_logo.jpg">logo ETI</image>
   <image source="img/pg_logo.jpg">logo PG</image>
</media>
```

13. (1,5pkt) W pliku XML do znacznika `study` dodać atrybut `kind`, który pozwoli rozróżnić pomiędzy laboratorium, wykładem i projektem. W pliku xsd dodać odpowiednią definicję. Dla zadeklarowanego atrybutu zdefiniować typ globalny i wykorzystując `enumeration` określić dopuszczalne, możliwe wartości atrybutu. Atrybut ma być wymagany. Uzupełnić plik XML o dane dotyczące laboratorium i projektu analogicznie jak zrobiono to dla wykładu.
14. **Poprosić prowadzącego o sprawdzenie pracy**
15. (1pkt) Stworzyć typ globalny prosty `shortStringType` oparty o typ łańcucha znaków. Określić maksymalną dopuszczalną długość łańcucha znaków na 30. Wykorzystać zdefiniowany typ w deklaracji elementów: `component, name`. Analogicznie stworzyć typ `longStringType` o długości 50 znaków i wykorzystać w wybranych miejscach (np. `surname`, ….)
16. **Poprosić prowadzącego o sprawdzenie pracy**
17. (0,5pkt) W elemencie `study` umożliwić opcjonalne umieszczanie atrybutu `obligatory`. Jego wartość może przyjmować tylko wartości „yes” lub „no” (wykorzystać `pattern`). Typ dla atrybutu zdefiniować lokalnie. Ustawić dla atrybutu wartość domyślną na „no”. Dla elementu `study` związanego z laboratorium ustawić w pliku XML wartość atrybutu na „yes”
18. (1pkt) W elemencie `text` wyszukać zdanie „Hypertext, in other words!” i umieścić je w znaczniku `subtitle`. Tak zmienić deklarację elementu `text`, aby plik się walidował.
19. **Poprosić prowadzącego o sprawdzenie pracy**
20. (2pkt) W pliku XML dodać fragment hierarchii: element z trzema podelementami, do jednego podelementu dodać atrybut. Dla jednego z podelementów wykorzystać nowy, zdefiniowany lokalnie typ, dla drugiego nowy, zdefiniowany globalnie typ. Trzeci podelement ma być wybierany z dwóch określonych elementów (`choice`). Dodawany fragment powinien być związany tematycznie z danymi zawartymi w pozostałych elementach.

XML i XML Schema - krótka ściąga ☺

XML

- wszystkie niepuste elementy muszą mieć znacznik początkowy i końcowy
- elementy mogą być zagnieżdżone, nie mogą na siebie zachodzić
- rozróżnianie dużych i małych liter

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<pajeczaki xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="Zad1_Pajeczaki.xsd">
    <pajaki gatunek="tygrzyk" chroniony="tak">
        <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
        <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
        <gromada>Pajeczaki</gromada>
    </pajaki>
    <pajaki gatunek="krzyzak" chroniony="nie">
        <nazwa jezyk="polska">Krzyżak Ogrodowy</nazwa>
        <nazwa jezyk="łacińska">Araneus diadematus</nazwa>
        <gromada>Pajeczaki</gromada>
    </pajaki>
</pajeczaki>
```

deklaracja XML

zawartość tekstowa elementu

atrybut

wartość atrybutu

znacznik początkowy

znacznik końcowy

korzeń (element główny)

XML Schema

Jeśli chcemy stworzyć:

- tylko element z zawartością tekstową
  - typ prosty
- element z podelementami
  - typ złożony
- element z podelementami i atrybutami
  - typ złożony
- element z zawartością mieszaną (podelementy i tekst)
  - typ złożony z atrybutem mixed=true
- element z atrybutem i zawartością tekstową
  - typ złożony z zawartością prostą (complexType simpleContent)

1) Definicja typu prostego nazwanego

```
<xs:simpleType name=”krotki_string”>
    < xs:restriction base=“string”/>
        < xs:maxLength value=“20” />
    </ xs:restriction>
</ xs:simpleType>
```

typ prosty nazwany

ograniczenie

2) Definicja elementu

```
<xs:element name="pajaki" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="nazwa" maxOccurs="unbounded">
        <xs:complexType mixed="true">
          <xs:attribute name="jezyk" type="xs:string" />
        </xs:complexType>
      </xs:element>
      <xs:element name="gromada" type="xs:string"/>
    </xs:sequence>
    <xs:attribute name="gatunek" type="xs:string" />
    <xs:attribute name="chroniony" type="xs:string" />
  </xs:complexType>
</xs:element>
```

liczba wystąpień

definicja elementu

typ złożony, lokalny

sekwencja, elementy w ściśle określonej kolejności

typ złożony z zawartością mieszaną

typ atrybutu

deklaracja atrybutu (zawsze po deklaracjach elementów)

3) Wyliczenia - lista predefiniowanych wartości

```
<xs:simpleType name=“nazwa_typu” >
  < xs:restriction base=“string”>
    < xs:enumeration value=“wartosc1” />
    < xs:enumeration value=“ wartosc2” />
    < xs:enumeration value=“ wartosc3” />
  </ xs:restriction>
</ xs:simpleType>
```

4) SimpleContent

Gdy tworzymy pochodny typ złożony na podstawie typu prostego lub innego typu złożonego o zawartości prostej. Można w ten sposób np. dodać atrybuty do prostego typu bazowego.

```
<xs:complexType name="nazwa-typu">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="nazwa_atrybutu" type="xs:string"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

5) Odniesienia do elementu

```
<xs:element name="data" type="xs:date"/>
```

globalna definicja elementu

```
<xs:element ref="data" minOccurs="0"/>
```

odniesienie do elementu zdefiniowanego globalnie