Задание за решения на задачи по Модул 3: XML валидиране чрез XML Schema

Изготвил: Марина Дабова, ФН: 62503, група 3, специалност СИ

Коментарите са обозначени с червен цвят

#### **Задача 1: Превърнете дадения по-долу DTD документ в XML Schema. Създайте XML екземпляр на тази схема и я валидирайте.**

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="collection"> <!-- елемента-корен е collection-->

<xs:complexType> <!-- дефинираме типа на collection

в случая е сложен тип, защото съдържа под-елементи-->

<xs:sequence> <!-- дефинираме последователността от неговите елементи -->

<xs:element ref="description"/>

<xs:element ref="recipe" minOccurs="0" maxOccurs="unbounded"/> <!-- тук задаваме честотата на срещане и тъй като

recipe има "\*" то се среща 0 или повече пъти и това е показано -->

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="description"> <!-- дефинираме си всеки под-елемент -->

<xs:complexType mixed="true">

<xs:sequence>

<xs:any minOccurs="0" maxOccurs="unbounded"/> <!-- използваме any,защото в DTD граматиката

description е използвано с ANY,което значи, че е елемент с произволно съдържание-->

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="recipe">

<xs:complexType>

<xs:sequence>

<xs:element name="title" type="string" />

<xs:element ref="ingredient" minOccurs="0" maxOccurs="unbounded" /> <!-- тук задаваме честотата на срещане и тъй като

ingredient има "\*" то се среща 0 или повече пъти и това е показано -->

<xs:element ref="preparation" />

<xs:element name="comment" type="string" minOccurs="0" maxOccurs="1" /> <!-- тук задаваме честотата на срещане и тъй като

comment има "?" то се среща 0 или 1 пъти и това е показано -->

<xs:element ref="nutrition" content="empty" />

<xs:/sequence>

<xs:/complexType>

</xs:element>

<xs:element name="ingredient">

<xs:complexType>

<xs:sequence>

<xs:element ref="ingredient" minOccurs="0" maxOccurs="unbounded"/>

<xs:element ref="preparation" minOccurs="0" maxOccurs="1"/> <!-- тук задаваме честотата на срещане и тъй като

preparation има "\*" то се среща 0 или 1 пъти и това е показано -->

</xs:sequence>

<xs:attribute name="name" type="string" use="required"/> <!--задаваме use="required", защотов DTD граматиката имаме, че атрибута е задължителен -->

<xs:attribute name="amount" type="string"/> <!-- не задаваме use, защото в DTD граматиката имаме, че атрибута е IMMPLIED -->

<xs:attribute name="unit" type="string"/> <!-- не задаваме use, защото в DTD граматиката имаме, че атрибута е IMMPLIED -->

</xs:complexType>

</xs:element>

<xs:element name="preparation">

<xs:complexType>

<xs:sequence >

<xs:element ref="step" type="string" minOccurs="0" maxOccurs="unbounded"/> <!-- тук задаваме честотата на срещане и тъй като

step има "\*" то се среща 0 или повече пъти и това е показано -->

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="nutrition">

<xs:complexType>

<xs:attribute name="protein" use="required"/> <!--задаваме use="required", защотов DTD граматиката имаме, че атрибута е задължителен -->

<xs:attribute name="carbohydrates" use="required"/> <!--задаваме use="required", защотов DTD граматиката имаме, че атрибута е задължителен -->

<xs:attribute name="fat" use="required"/> <!--задаваме use="required", защотов DTD граматиката имаме, че атрибута е задължителен -->

<xs:attribute name="calories" use="required"/> <!--задаваме use="required", защотов DTD граматиката имаме, че атрибута е задължителен -->

<xs:attribute name="alcohol"/> <!-- не задаваме use, защотов DTD граматиката имаме, че атрибута е IMMPLIED -->

</xs:complexType>

</xs:element>

</element>

</xs:schema>

<!-- xml екземляр -->

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

<collection xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="zad2.xsd">

<description> </description>

<recipe>

<title> </title>

<ingredient name=""

amount =""

unit="">

<preparation>

<step> </step>

<step> </step>

</preparation>

</ingredient>

<nutrition protein=""

carbohydrates=""

fat=""

calories=""

alcohol="">

</nutrition>

<comment> </comment>

</recipe>

</collection>

#### **Задача 2: Даденият по-долу XML документ описва типовете сметки, които поддържа една примерна банка, нейните клиенти и сметките, които те имат. За този XML документ създайте XML Schema, която изпълнява следните условия:**

* Редът на срещане на под-елементите на bank (accounts, customers и customer\_accounts) и accounts (saving\_accounts и checking\_accounts) няма значение
* Всяка сметка има уникален идентификатор
* Всеки клиент има уникален идентификатор
* Балансът на сметката не може да бъде по-малък от -5000 - за това условие използвайте рестрикция на съществуващите предефинирани типове
* Атрибутът c\_id реферира към съответния клиент, а ac\_id към съответната му сметка
* Дефинирайте елементите customers и saving\_account като комплексен глобален тип, а елемента accounts като комплексен локален тип

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="bank"> <!--елемента-корен -->

<xs:complexType> <!-- дефинираме под-елементите -->

<xs:all> <!--реда е без значение -->

<xs:element ref="accounts"/> <!--accounts трябва да е дефиниран локално -->

<xs:complexType>

<xs:all>

<xs:element ref="saving\_accounts"/>

<xs:element ref="checking\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element ref="customers"/>

<xs:element ref="customer\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element name="saving\_accounts"> <!--saving\_accounts трябва да е дефиниран глобално -->

<xs:complexType>

<xs:element ref="saving\_account" />

</xs:complexType>

</xs:element>

<xs:element name="saving\_account">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/> <!--има уникален идентификатор -->

<xs:attribute name="interest" type="decimal" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element ref="checking\_account" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/> <!--има уникален идентификатор -->

</xs:complexType>

</xs:element>

<xs:element name="balance"> <!--балансът на сметката те може да е по-малък от -5000 -->

<xs:simpleType>

<xs:restriction base="integer">

<xs:minInclusive value="-5000"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element name="customers"> <!--customers трябва да е дефиниран глобално -->

<xs:complexType>

<xs:element ref="customer" />

</xs:complexType>

</xs:element>

<xs:element name="customer">

<xs:complexType>

<xs:element ref="name" type="string"/>

<xs:element ref="address"type="string" />

<xs:attribute name="id" use="required" type="ID"/> <!--има уникален идентификатор -->

</xs:complexType>

</xs:element>

<xs:element name="customer\_accounts"> <!--customer\_accounts трябва да е дефиниран глобално -->

<xs:complexType>

<xs:element ref="customer\_account"/>

</xs:complexType>

</xs:element>

<xs:element name="customer\_account">

<xs:complexType>

<xs:attribute name="c\_id" use="required" type="IDREF"/> <!-- c\_id реферира към съответния клиент -->

<xs:attribute name="ac\_id" use="required" type="IDREF"/> <!-- ac\_id реферира към съответната сметка -->

</xs:complexType>

</xs:element>

</xs:schema>

#### **Задача 3: В XML схемата от задача 2 направете следните промени:**

* Дефинирайте два прости типа, задаващи горна и долна граница на стойността на елемента balance. След това променете дефиницията на елемента balance, като го представите като обединение на тези два прости типа
* Използвайки разширени типове на XML Schema, създайте нов тип customerExt, който разширява дефиницията на типа customer като добавя нов негов под-елемент contacts, който от своя страна се състои от 2 под-елемента - email и telephone. Използвайте новия тип customerExt вместо customer и запишете новата XML схема под името bank.xsd
* Създайте нова XML схема, която предефинира типа balance и customerExt от външната за нея XML схема - bank.xsd. Новият тип balance не трябва да бъде по-малък от -5200,a новият тип customerExt съдържа допълнително нов под-елемент image от тип base64Binary, който има един атрибут src от тип string. Запишете новата XML схема под името bankExt.xsd
* Добавете няколко анотации в по-горе създадената XML схема (bank.xsd) към избрани от вас комплексните типове, описващи тяхното предназначение

Това е файла bank.xsd

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="bank">

<xs:complexType>

<xs:all>

<xs:element ref="accounts"/>

<xs:complexType>

<xs:all>

<xs:element ref="saving\_accounts"/>

<xs:element ref="checking\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element ref="customers"/>

<xs:element ref="customer\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element name="saving\_accounts">

<xs:complexType>

<xs:element ref="saving\_account" />

</xs:complexType>

</xs:element>

<xs:element name="saving\_account">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/>

<xs:attribute name="interest" type="decimal" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element ref="checking\_account" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/>

</xs:complexType>

</xs:element>

<xs:element name="balance">

<xs:simpleType >

<xs:union memberTypes="minBalance maxBalance"/> <!-- променяме дефиницията на balance като го представим като обединение на границите-->

</xs:simpleType>

</xs:simpleType>

<xs:simpleType name="minBalance"> <!-- задаваме долна граница за balance-->

<xs:restriction base="negativeInteger">

<xs:minInclusive value="-5000"/>

</xs:restriction>

<xs:simpleType name="maxBalance"> <!-- задаваме горна граница за balance-->

<xs:restriction base="positiveInteger">

<xs:maxInclusive value="1000000"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element name="customer">

<xs:complexType>

<xs:element ref="name" type="string"/>

<xs:element ref="address"type="string" />

<xs:attribute name="id" use="required" type="ID"/>

</xs:complexType>

</xs:element>

<xs:element name="customer" type="customer"/>

<xs:complexType name="customerExt"> <!-- добавя нов елемент customerExt-->

<xs:complexContent>

<xs:extension base="customer"> <!-- customerExt разширява customer-->

<xs:sequence>

<xs:element ref="contacts" /> <!--customerExt има под-елемент contacts-->

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="customerExt" type="customerExt"/>

<xs:complexType name="contacts">

<xs:sequence>

<xs:element ref="email"/> <!--contacts има под-елемент email-->

<xs:element ref="telephone"/> <!--contacts има под-елемент telephone-->

</xs:sequence>

</xs:complexType>

<xs:element name="contacts" type="contacts"/>

<xs:element name="email" type="string"/>

<xs:element name="telephone" type="string"/>

<xs:complexType name="customers">

<xs:sequence>

<xs:element ref="customerExt" /> <!-- използваме новия тип customerExt вместо customer -->

</xs:sequence>

</xs:complexType>

<xs:element name="customer\_accounts">

<xs:complexType>

<xs:element ref="customer\_account"/>

</xs:complexType>

</xs:element>

<xs:element name="customer\_account">

<xs:complexType>

<xs:attribute name="c\_id" use="required" type="IDREF"/>

<xs:attribute name="ac\_id" use="required" type="IDREF"/>

</xs:complexType>

</xs:element>

</xs:schema>

Това е файла bankExt.xsd

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:redefine schemaLocation="bank.xsd">

<xs:simpleType name="minBalance">

<xs:restriction base="minBalance">

<xs:minInclusive value="-5200"/> <!-- задаваме минималната граница на balance вече да е -5200 -->

</xs:restriction>

</xs:simpleType>

<xs:complexType name="customerExt">

<xs:complexContent>

<xs:extension base="customerExt">

<xs:sequence>

<xs:element name="image" type="base64Binary"/> <!-- имаме нов под-елемент image от тип base64Binary -->

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:redefine>

<xs:complexType name="image">

<xs:simpleContent>

<xs:extension base="base64Binary">

<xs:attribute name="src" type="string" use="required"/> <!-- новия под-елемент image има един атрибут src от тип string -->

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:schema>

#### **Задача 4: Редактирайте XML схемата от задача 3 (bank.xsd), като добавите елемент postalCode към комплексния тип customer и включите регулярен израз за:**

* пощенски код (четири цифрено число, например:1000)
* телефон (например в следния формат:+359-02-989-14-04)
* електронна поща

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="bank">

<xs:complexType>

<xs:all>

<xs:element ref="accounts"/>

<xs:complexType>

<xs:all>

<xs:element ref="saving\_accounts"/>

<xs:element ref="checking\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element ref="customers"/>

<xs:element ref="customer\_accounts"/>

</xs:all>

</xs:complexType>

</xs:element>

<xs:element name="saving\_accounts">

<xs:complexType>

<xs:element ref="saving\_account" />

</xs:complexType>

</xs:element>

<xs:element name="saving\_account">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/>

<xs:attribute name="interest" type="decimal" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element ref="checking\_account" />

</xs:complexType>

</xs:element>

<xs:element name="checking\_accounts">

<xs:complexType>

<xs:element name="balance" type="integer" />

<xs:attribute name="id" use="required" type="ID"/>

</xs:complexType>

</xs:element>

<xs:element name="balance">

<xs:simpleType >

<xs:union memberTypes="minBalance maxBalance"/>

</xs:simpleType>

</xs:simpleType>

<xs:simpleType name="minBalance">

<xs:restriction base="negativeInteger">

<xs:minInclusive value="-5000"/>

</xs:restriction>

<xs:simpleType name="maxBalance">

<xs:restriction base="positiveInteger">

<xs:maxInclusive value="5000"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element name="customer">

<xs:complexType>

<xs:element ref="name" type="string"/>

<xs:element ref="address"type="string" />

<xs:element ref="postalCode"/> <!—добавяме нов елемемент към customer- postalCode-->

<xs:attribute name="id" use="required" type="ID"/>

</xs:complexType>

</xs:element>

<xs:element name="customer" type="customer"/>

<xs:complexType name="customerExt">

<xs:complexContent>

<xs:extension base="customer">

<xs:sequence>

<xs:element ref="contacts" />

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

<xs:element name="customerExt" type="customerExt"/>

<xs:complexType name="contacts">

<xs:sequence>

<xs:element ref="email"/>

<xs:element ref="telephone"/>

</xs:sequence>

</xs:complexType>

<xs:element name="contacts" type="contacts"/>

<xs:element name="email" type="string"/>

<xs:element name="telephone" type="string"/>

<xs:complexType name="customers">

<xs:sequence>

<xs:element ref="customerExt" />

</xs:sequence>

</xs:complexType>

<xs:element name="customer\_accounts">

<xs:complexType>

<xs:element ref="customer\_account"/>

</xs:complexType>

</xs:element>

<xs:element name="customer\_account">

<xs:complexType>

<xs:attribute name="c\_id" use="required" type="IDREF"/>

<xs:attribute name="ac\_id" use="required" type="IDREF"/>

</xs:complexType>

</xs:element>

<xs:simpleType name="postalCode">

<xs:restriction base="integer">

<xs:pattern value="[0-9]{4}"/> <!-- създаваме регулярен израз за postalCode като имаме, че трябва да е четирицифрено число:

[0-9]посочва,че числата ще са от 0 до 9,а {4} -че числото ще се повтори 4 пъти -->

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="telephone">

<xs:restriction base="string">

<xs:pattern value="\+359-[0-9]{2}-[0-9]{3}-[0-9]{2}-[0-9]{2}"/> <!-- създаваме регулярен израз за telephone с формат :+359-02-989-14-04 -->

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="email">

<xs:restriction base="token">

<xs:pattern value=""[a-zA-Z0-9]+@+[a-z]+[.]+[a-z]"/> <!-- създаваме регулярен израз за email -->

</xs:restriction>

</xs:simpleType>

</xs:schema>

Използваме синтаксис за използване на регулярен израз:  
<xsd:restriction base="XXXX">  
   <xsd:pattern value="Regular\_Expression"/>  
</xsd:restriction>

#### **Задача 5: Редактирайте XML схемата от задача 3 (bankExt.xsd), като добавите към елемента image атрибут type от тип нотация. Създайте XML инстанция на новата схема и валидирайте.**

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:redefine schemaLocation="bank.xsd">

<xs:simpleType name="minBalance">

<xs:restriction base="minBalance">

<xs:minInclusive value="-5200"/>

</xs:restriction>

</xs:simpleType>

<xs:complexType name="customerExt">

<xs:complexContent>

<xs:extension base="customerExt">

<xs:sequence>

<xs:element name="image" />

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

</xs:redefine>

<xs:notation name="png" public="image/png" system="png.com"/> <!--Добавяме нотация -->

<xs:complexType name="image">

<xs:simpleContent>

<xs:extension base="base64Binary">

<xs:attribute name="src" type="string" use="required"/>

<xs:attribute name="type" use="required"> <!--Добавяме елемент type от тип нотация -->

<xs:simpleType>

<xs:restriction base="NOTATION"> <!--Добавяме нотация -->

<xs:enumeration value="png"/>

</xs:restriction>

</xs:simpleType>

</xs:attribute>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

Използваме синтаксис за дефиниране на нотации в XML схема: <notation id=ID name=NCName public=anyURI system=anyURI any attributes>(annotation?)</notation>

#### **Задача 6: Съставете XML Schema, която включва схеми от други пространства от имена. Създайте XML екземляр (инстанция) на тази схема и го валидирайте.**

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

<collection xmlns:xsi="http://www.w3.org/2001/XMLSchema" xsi:noNamespaceSchemaLocation="zad6.xsd">

<description> </description>

<recipe>

<title> </title>

<ingredient name=""

amount =""

unit="">

<preparation>

<step> </step>

<step> </step>

</preparation>

</ingredient>

<nutrition protein=""

carbohydrates=""

fat=""

calories=""

alcohol="">

</nutrition>

<comment> </comment>

</recipe>

</collection>

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">

<xs:element name="collection">

<xs:complexType>

<xs:sequence>

<xs:element ref="description"/>

<xs:element ref="recipe" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="description">

<xs:complexType mixed="true">

<xs:sequence>

<xs:any minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="recipe">

<xs:complexType>

<xs:sequence>

<xs:element name="title" type="string" />

<xs:element ref="ingredient" minOccurs="0" maxOccurs="unbounded" />

<xs:element ref="preparation" />

<xs:element name="comment" type="string" minOccurs="0" maxOccurs="1" />

<xs:element ref="nutrition" content="empty" />

<xs:/sequence>

<xs:/complexType>

</xs:element>

<xs:element name="ingredient">

<xs:complexType>

<xs:sequence>

<xs:element ref="ingredient" minOccurs="0" maxOccurs="unbounded"/>

<xs:element ref="preparation" minOccurs="0" maxOccurs="1"/>

</xs:sequence>

<xs:attribute name="name" type="string" use="required"/>

<xs:attribute name="amount" type="string"/>

<xs:attribute name="unit" type="string"/>

</xs:complexType>

</xs:element>

<xs:element name="preparation">

<xs:complexType>

<xs:sequence >

<xs:element ref="step" type="string" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="nutrition">

<xs:complexType>

<xs:attribute name="protein" use="required"/>

<xs:attribute name="carbohydrates" use="required"/>

<xs:attribute name="fat" use="required"/>

<xs:attribute name="calories" use="required"/>

<xs:attribute name="alcohol"/>

</xs:complexType>

</xs:element>

</element>

</xs:schema>

zad6.xsd:

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

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:num="http://example.com">

<xsd:import namespace="http://example.com" schemaLocation="zad6.xsd"/> <!--включваме схема от друго пространство от имена -->

<xsd:element name="collection" type="collectionType"/>

<xsd:complexType name="collectionType">

<xsd:sequence>

<xsd:element ref="num:recipe" maxOccurs="unbounded"/>

</xsd:sequence>

</xsd:complexType>

</xsd:schema>

Използваме синтаксиса: за импортиране на външна схема: <import id=ID namespace=anyURI schemaLocation=anyURI any attributes >(annotation?)</import> като в случая при нас е - xsd:import namespace="http://example.com" schemaLocation="zad6.xsd"/>