**Ministério da Educação da Federação Russa**

**Agência Federal de Educação**

**Instituição estadual de ensino superior**

**UNIVERSIDADE TÉCNICA DO ESTADO DE NIZHNY NOVGOROD**

**eles. RÉ. Alekseeva**

**Departamento de Informática e Sistemas de Controle**

**Gestão de dados**

**Introdução ao XML**

Orientações para trabalhos laboratoriais nº 6 para alunos da direção 09.03.01 "Informática e Engenharia de Computadores", 09.03.02 "Sistemas e Tecnologias de Informação"

**Nizhny Novgorod**

**2020**

Compilado por: T.I. Balashova

UDC

Base de dados

NSTU; Comp.: T.I. Balashova

N. Novgorod 2020. 13 p.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Universidade Técnica do Estado de Nizhny Novgorod.

Casa de impressão de NSTU. 603950, Nizhny Novgorod, rua. Minina, 24.

**linguagem de marcação XML**

O meio-termo entre SGML e HTML é XML (eXtensible Markup Language), uma linguagem de marcação de documentos que originalmente surgiu como uma tecnologia para compartilhamento de documentos na World Wide Web, mas evoluiu gradualmente para um meio de armazenamento e transmissão de estruturas hierárquicas de dados. É um subconjunto da linguagem **SGML** . *padrão generalizado marcação Language* - uma linguagem de marcação generalizada padrão) - uma metalinguagem na qual você pode definir a linguagem de marcação para documentos (Fig. 1). XML apareceu em 1986.

**GML**

**SGML**

**HTML**

**XML**

**XHTML**

**…...**

Arroz. 1. Um subconjunto da linguagem GML

**Um modelo de dados hierárquico** é um modelo de dados no qual os dados são representados em uma estrutura hierárquica (árvore). Um banco de dados hierárquico consiste em entidades que podem ter entidades pai e filho, mas sempre há uma entidade de documento - o elemento raiz, do qual tudo começa e que não possui um elemento pai, e todos os outros elementos são seus descendentes (Fig. 2 ). Bancos de dados hierárquicos são otimizados para leitura de informações.

Корневой элемент

Потомок

Потомок

Потомок

Потомок

Потомок

Потомок

Потомок

Потомок

Потомок

Arroz. 2. Estrutura hierárquica de dados

Um documento XML consiste em entidades que podem conter dados de caracteres e elementos de marcação. A marcação inclui tags (tags em inglês) - linhas de texto entre colchetes angulares ( *< >* ) e denotando os limites dos elementos, bem como atributos de elemento (atributos em inglês) e comentários. A linguagem XML não contém um conjunto de tags como o HTML, mas descreve as regras para criar suas próprias tags. Um documento *bem formado* é um documento XML escrito com todas as regras em mente.

No início do documento XML, o elemento raiz é precedido por um prólogo. O prólogo deve começar (para versões da linguagem anteriores a 1.0) com uma declaração XML, por exemplo:

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

Atributo (versão="1.0") - versão XML, e o atributo (encoding="UTF-8") é a codificação do documento.

A instrução especial !DOCTYPE, que permite usar a linguagem de esquema DTD para determinar o esquema de um documento, ou seja, quais elementos e atributos os elementos podem ter em um determinado documento, em que quantidade e possíveis formas de “aninhamento” mútuo de elementos. O prólogo também pode conter instruções de processamento que permitem, por exemplo, vincular um documento XML a folhas de estilo CSS:

<?xml-stylesheet href="style.css"?>

Em qualquer lugar em um documento XML, os comentários podem começar com *<!--* e terminar com *-->.*

Um elemento não pode ter dois atributos idênticos (atributos com o mesmo nome). Os nomes de elementos e atributos devem começar com uma letra, um sublinhado ( \_ ) ou dois pontos (:). Eles podem

contêm apenas letras Unicode, algarismos arábicos, hífens, sublinhados, pontos e dois-pontos. Os nomes não podem começar com a string "xml" em nenhum caso.

Existem regras que devem ser observadas para a correta construção de documentos XML:

1.O documento XML começa com uma declaração XML (para versões de idioma anteriores a 1.0).

2.O documento XML contém um elemento raiz.

3.Em um documento XML, os comentários começam com **<!--** e terminam com **-->.**

4.Em um documento XML, nomes de elementos e nomes de atributos não podem conter espaços. Os nomes dos elementos diferenciam maiúsculas de minúsculas.

5. Em um documento XML, todas as tags abertas devem ser estar fechado.

6. Em um documento XML não é permitida a sobreposição de elementos, ou seja, é necessário observar os níveis de aninhamento dos elementos.

Em um documento XML, o conteúdo do elemento ou o valor do atributo pode conter caracteres especiais. Esses caracteres podem, em alguns casos, fazer parte dos dados do caractere e, em alguns casos, podem fazer parte das construções incluídas na sintaxe XML. Para que os programas que processam um documento XML entendam corretamente o que fazer com caracteres especiais, eles devem ser escapados (Fig. 3)

Ссылка на сущность:

& &amp;

< &It;

> &gt;

` &apos;

“ &qout;

Неразрывный пробел –

Ссылка по номеру символа:

& &#38;

< &#60;

> &#62;

` &#39;

“ &#34;

Неразрывный пробел &#160;

Экранирование специальных символов в XML

Arroz. 3. Escapando caracteres especiais em XML



Arroz. 4. Um exemplo de compilação de um documento XML.

Уровни правильности

XML-документа

корректные

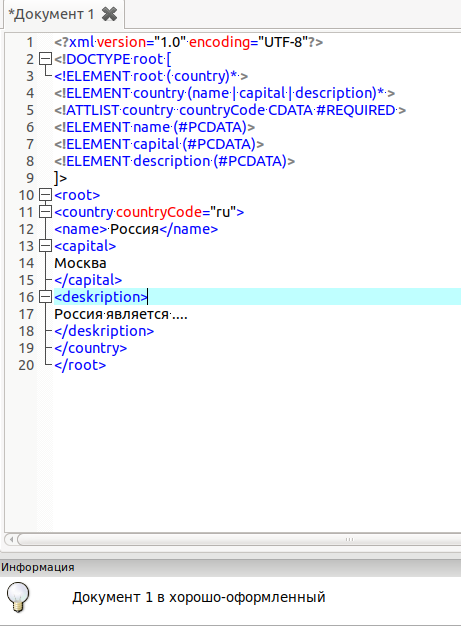
валидные

Fig.5. Níveis de validade do documento XML

Um documento XML válido é um documento XML válido que atende a requisitos adicionais para sua estrutura interna, descritos usando linguagens de esquema como DTD ou XML Schema.

DTD (Document Type Definition) permite especificar a estrutura e o conjunto permitido de elementos em um documento XML e seus atributos. Se você usar um documento XML como um banco de dados hierárquico, poderá usar um DTD para descrever a estrutura desse banco de dados. Os aplicativos podem usar esquemas DTD para validar (validar) vários arquivos XML contra eles.

Um esquema DTD pode ser declarado dentro e fora de um documento XML. Para declarar um DTD dentro de um documento XML, use a diretiva <!DOCTYPE>, localizada abaixo da declaração XML. Dentro da diretiva <!DOCTYPE>, o elemento XML raiz e a descrição do esquema entre colchetes são indicados (Fig. 6)



Arroz. 5. Um exemplo de compilação de uma definição de tipo de documento XML Documento DTD.

Código do esquema DTD mostrado na Figura 5:

1. !DOCTYPE root especifica que o elemento raiz deste documento XML é o elemento raiz ;
2. !ELEMENT root ( country ) - especifica que o elemento raiz deve conter um elemento aninhado: country ;
3. ! ELEMENTO país ( nome | capital | descrição ) - especifica que o elemento país deve contém três elementos aninhados: name , capital , deskription ;
4. ! ATTLIST país Código do país CDATA # REQUIRED - declaração do atributo countryCode , # REQUIRED - indica a presença obrigatória deste atributo
5. ! ELEMENTO name especifica que o elemento name contém dados do tipo #PCDATA;
6. !ELEMENT capital - especifica que o elemento capital contém dados do tipo #PCDATA;
7. !ELEMENT descrição especifica que o elemento descrição contém dados do tipo #PCDATA;

Ao usar uma declaração DTD externa, a diretiva <!DOCTYPE> deve conter um link para o arquivo DTD, por exemplo, através do ID do sistema: <! DOCTYPE raiz SISTEMA " raiz . dtd "> . Neste código: root é o nome do elemento raiz, SYSTEM é uma indicação de que o arquivo DDT está localizado no computador usuário, " root .dtd" é o nome do arquivo DTD (se especificado apenas o nome do arquivo sem o caminho, então este arquivo localizado na mesma pasta que o documento XML).

Link para DTD externo via identificador público do seguinte modo:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

"http://www.w3.org/TR/html4/strict.dtd">

O próprio arquivo DTD deve conter uma descrição de esquema que, quando declaração interna é colocada dentro de colchetes na diretiva <!DOCTYPE>. Declarando elementos em um DTD

O seguinte é usado para declarar elementos em um DTD sintaxe:

<!ELEMENT element\_name categoria>

ou

<!ELEMENT nome\_do\_elemento (conteúdo do elemento)>

Um elemento vazio é declarado com uma palavra-chave vazio,

indicando sua categoria:

<!ELEMENT element\_name VAZIO>

Alguns elementos podem conter apenas processados dados de caracteres (inglês PCDATA - dados de caracteres analisados). Esses os dados são colocados entre as tags inicial e final. Esses os dados não podem incluir caracteres &, < ou >. Esses símbolos deve ser escapado, por exemplo, com referências a entidades &amp; &lt; e &gt; respectivamente.

Se o elemento contiver apenas caracteres analisáveis data, é declarado especificando a palavra-chave #PCDATA entre parênteses indicando o conteúdo do elemento:

<!ELEMENT element\_name (#PCDATA)>

Um elemento com conteúdo arbitrário é declarado com usando a palavra-chave ANY para indicar sua categoria:

<!ELEMENT element\_name ANY>

A palavra-chave ANY indica que o elemento pode conter dados de caracteres e quaisquer dados aninhados elementos. Se o elemento contiver elementos filhos, seus nomes são declarados como conteúdo de elemento dentro da rodada colchetes:

<!ELEMENT element\_name (child\_element\_name\_1)>

Ou

<!ELEMENT element\_name (child\_element\_name\_1,

child\_element\_name\_2, child\_element\_name\_3)>

O número de elementos filho não é limitado. Subsidiárias elementos separados por vírgulas devem seguir em elemento pai exatamente na ordem em que são especificado no DTD.

Se o nome do elemento filho for especificado sem nenhum caracteres especiais no final, como mostrado no exemplo acima, isso significa que o elemento pai deve conter exatamente um tal elemento filho. Portanto, o código <!ELEMENT root(child1, child2, child3)> significa dentro do elemento raiz deve conter exatamente três elementos filhos chamados filho1, filho2 e filho3 nessa ordem.

Os seguintes caracteres especiais existem para determinar o número de ocorrências possíveis de um elemento filho:

\* - qualquer número de elementos filhos (zero ou mais);

+ - pelo menos um elemento filho (um e mais);

? - presença opcional de um elemento filho (zero ou um);

A seguir está um exemplo de declaração de um elemento filho, que deve aparecer no pai pelo menos uma vez:

<!ELEMENT element\_name (child\_element\_name\_1+)>

Há também um caractere especial | (Barra vertical), permitindo especificar a separação de alternativas, por exemplo:

<!ELEMENT element\_name (child\_element\_1| child\_element\_2>)

Este código declara que dentro deste elemento, conter um elemento child\_element\_1 ou um elemento filha\_el\_2.

Declarações de elemento filho:

<! ELEMENTO root (# PCDATA | nome | maiúscula | descrição )\*>

Este código declara que dentro do elemento raiz, contém elementos aninhados name , capital , deskription , ou dados de personagem processados, e em qualquer pedido e quantidade.

Declarando atributos em um DTD

A seguinte sintaxe é usada para declarar atributos:

<!ATTLIST element\_name attribute\_name attribute\_type attribute\_property>

Desvantagens dos DTDs:

* Em DTD usa uma sintaxe de linguagem diferente de XML, que pode levar a vários erros, por exemplo, problemas com codificação.
* Não há verificação de tipo de dados no DTD (string, integer, float, boolean dados, data e hora, etc.), apenas um tipo de dados é usado - linha.
* Não há namespaces em um DTD, então você não pode mapear dois ou mais esquemas DTD para um documento XML.

Para eliminar essas deficiências, foi desenvolvida a linguagem XML Schema.

**Parte 1**

Tarefa para a primeira parte.

Crie seu próprio documento XML válido que descreva qualquer banco de dados hierárquico. O documento deve conter vários tipos diferentes de elementos aninhados e pelo menos quatro níveis de aninhamento. Os elementos devem conter vários atributos e texto

valores. Usando o esquema de banco de dados do primeiro laboratório para compor um documento XML.

O relatório deve conter: uma listagem do documento XML, uma descrição dos elementos, uma conclusão sobre o trabalho realizado.

Perguntas de controle

1. O que é XML?
2. O que é um modelo de dados hierárquico?
3. O que podem conter as entidades que compõem um documento XML?
4. Para que são usadas as tags em documentos XML?
5. O que vem antes do elemento raiz de um documento XML?
6. Quais são as instruções usadas em documentos XML? em processamento?
7. O que são tags de início e fim?
8. Qual pode ser o nome de um elemento ou atributo em um documento XML?
9. Nomeie as regras para construir documentos XML válidos.
10. Como os comentários são marcados em um documento XML?
11. Tags não fechadas são permitidas em um documento XML?
12. O que são caracteres especiais em documentos XML?
13. Como os caracteres especiais podem ter escape em documentos XML?
14. O que é DTD?
15. O que é validação de documento XML?
16. Qual diretiva é usada para declarar um DTD dentro de um XML-

documento?

1. Quais são as desvantagens da linguagem DTD?

**Parte 2**

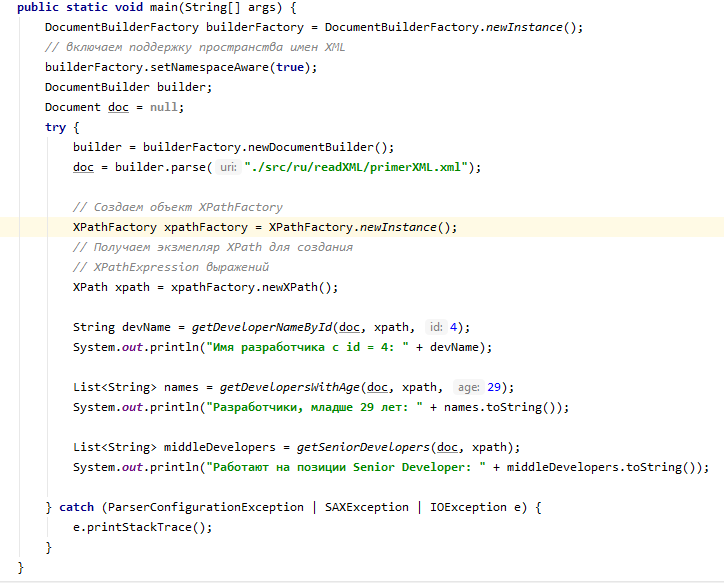
**parte teórica**

A linguagem de consulta XML é usada para acessar elementos documento XML. O resultado da avaliação de uma expressão XPath (solicitação) é uma indicação de uma área ou áreas específicas documento. Um documento XML é uma árvore cuja raiz é é o elemento raiz. Ramos saindo da raiz terminando com nós (eng. nodes) - elementos, seus atributos e conteúdo de texto. Cada nó da parte inferior nível, pode haver apenas um ancestral direto, mas ao mesmo tempo Um elemento pode ter muitos filhos.

O atributo de um elemento ou nó de conteúdo de texto não pode tem descendentes.

A linguagem Xpath distingue entre os seguintes tipos de nós: root ( documento nó ), elemento ( elemento nó ), atributo ( atributo nó ), texto ( texto nó ) e um comentário ( comentário nó ). nó raiz contém não apenas o elemento raiz do documento, mas também todos os outros nós, incluindo instruções de processamento, namespaces, comentários e dados de texto. O nó raiz é contêiner para outros nós. Nomes de nó raiz e correspondência do elemento raiz do documento. Os nomes também combinam nós de elementos e nós de atributos com os nomes dos correspondentes elementos e atributos. Os nós de comentário são inseridos na árvore sem nome. Os nós de texto também não têm nome - eles contêm o texto encontrado no documento XML entre os elementos.

Fig.6. Um exemplo de criação de uma solicitação para um documento XML .



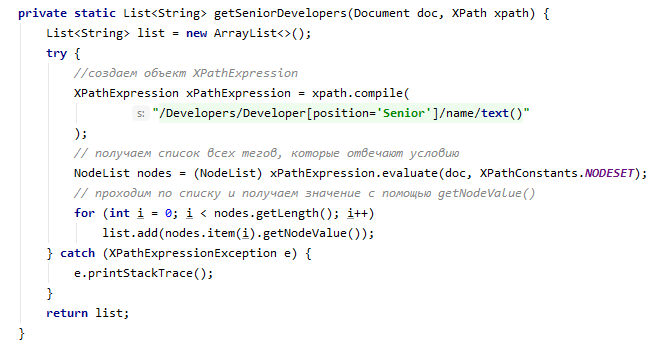


Fig.7. Um exemplo de obtenção de informações específicas de um documento XML .



Fig.8. O documento XML no qual a solicitação foi gravada.

Tarefa para a segunda parte.

Familiarize-se com a linguagem de consulta XPath. Escreva algumas consultas em seu documento XML criado anteriormente.

O relatório deve conter: uma listagem do programa, o resultado da saída do programa, uma conclusão sobre o trabalho realizado.