XML

Ненко Табаков
Пламен Танов
Технологическо училище “Електронни системи”
Технически университет – София

7 октомври 2008

ЛИТЕРАТУРА
НЕОБХОДИМИ ПРОГРАМИ

- W3C спецификация - http://www.w3.org/TR/2006/REC-xml11-20060816/
- Eclipse - www.eclipse.org
  - WTP – Web Tools Project

ВЪВЕДЕНИЕ

- XML (Extensible Markup Language) – описва клас от данни наречени XML документи и частично описва поведението на компютърни програми, които ги обработват
- XML е стеснена (ограничена) форма на SGML (Standard Generalized Markup Language). По конструкция XML документите са спазени SGML документи

ВЪВЕДЕНИЕ

- ***XML процесор*** – софтуерен модул, който се използва за четене на XML документи и предоставя достъп до тяхното съдържание и структура
- Предполага се, че ***XML процесор*** се използва от друг софтуерен модул наречен приложение

ИСТОРИЯ

XML е разработен от XML Working Group формирана под покровителството на World Wide Web Consortium (W3C) през 1996 и оглавена от Jon Bosac от Sun Microsystem и активно подпомагана от XML Special Interest Group.

ЦЕЛИ

1. ***XML*** да бъде прост за използване през Интернет
2. ***XML*** да има широко разпространение
3. ***XML*** да бъде съвместим с ***SGML***
4.Лесно да се пишат програми, които обработват ***XML*** документи
5. ***XML*** да бъде минимален и да няма нужда от разработване на допълнителни възможност (features)
6. ***XML*** документите трябва да са четими от хора и смислени по структура
7.Дизайнът на ***XML*** да се подготвя лесно
8.Дизайнът на ***XML*** да е съществен и кратък
9. ***XML*** документите да се създават лесно
10. ***XML*** *маркъп* да е максимално сбит и изразителен

XML ДОКУМЕНТ

- Един ***XML*** документ се състои от единици от данни наречени същности (entity)
- Една същност съдържа данни, които могат да бъдат както синтактично оформени (parsed data), така и данни които нямат синтактичен смисъл (unparsed data)
- Синтактично правилните данни се състоят от символи, някои от които формират символни данни, а други маркъпи
- ***XML*** предоставят механизъм за поставяне на ограничения върху данните и логическата структура на един документ

XML ДОКУМЕНТ

- ***XML*** документите могат да бъдат както *добре оформени,* така и *валидни*.
- Всеки ***XML*** документ има както логическа, така и физическа структура.
- Всеки документ има *корен*, още наричан документна същност
- Логическата структура на един документ се състои от декларации, елементи, коментари, препратки към символи и функции за обработка

ДОБРЕ ОФОРМЕН ***XML*** ДОКУМЕНТ

Добре оформеният ***XML*** документ изпълнява следните условия:

- съдържа един или повече елемента

```
Добре оформен документ :
<Name> Nenko Tabakov </Name>

Не добре оформен документ:
Nenko Tabakov
```

ДОБРЕ ОФОРМЕН *XML* ДОКУМЕНТ

- има само един елемент, който е *корен*, и никаква част от него не се появява в съдържанието на който и да е друг елемент

Добре оформен документ :

```
<PhoneBook>
   <Element>
      <Name> Nenko Tabakov </Name>
      <Phone>564389284 </Phone>
      <Email> fsajf@fhjsfs.com </Email>
      <Addres> fsfs, fshfds, fsdfs, fds 45 </Addres>
   </Element>
</PhoneBook>
```

Не добре оформен документ:

```
<Name> Nenko Tabakov </Name>
<Name> Plamen Tanov </Name>
<Phone>564389284 </Phone>
<Email> fsajf@fhjsfs.com </Email>
```

ДОБРЕ ОФОРМЕН *XML* ДОКУМЕНТ

- за всички други елементи ако отварящият таг е в съдържанието на друг елемент, затварящият таг е в съдържанието на същия елемент. По – просто казано, елементите, разделени с отварящ-затварящ таг са правилно вложени един в друг
- Всяка същност е добре оформена

Добре оформен документ :

```
<PhoneBook>
   <Element>
      <Name> Nenko Tabakov </Name>
   </Element>
</PhoneBook>
```

Не добре оформен документ:

```
<PhoneBook>
   <Element>
      <Name> Nenko Tabakov </Element>
   </Name>
</PhoneBook>
```

ДОБРЕ ОФОРМЕН ***XML*** ДОКУМЕНТ

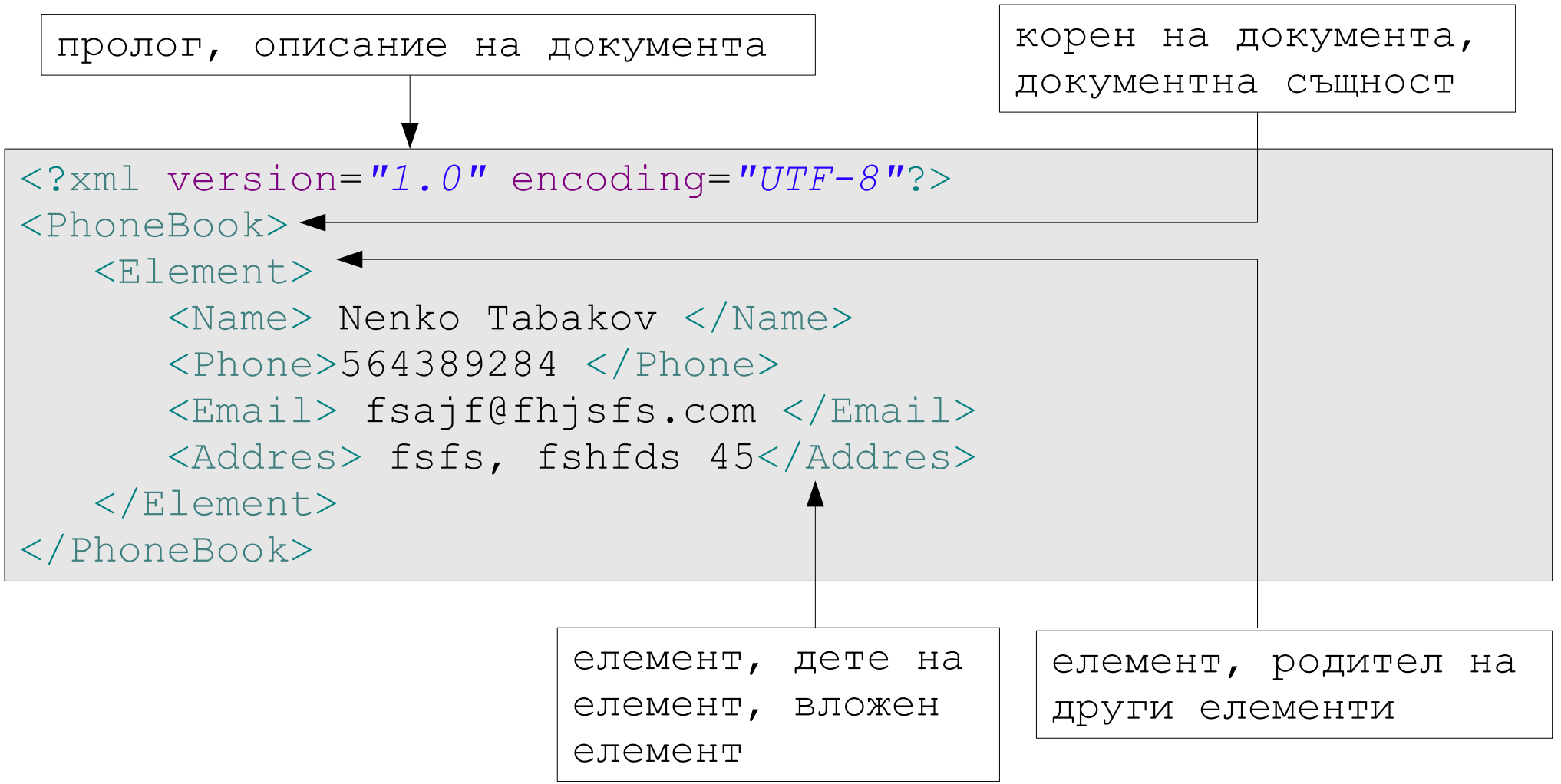
Елементът P се нарича родител на елемента C, ако елементът C е вложен в елементът P. От своя страна елементът C се нарича дете на елемента P

```
<P>
    <C></C>
</P>
```

ВАЛИДЕН *XML* ДОКУМЕНТ

- Един ***XML*** документ е валиден ако изпълнява предварително дефинирани условия
- Валиден ***XML*** документ е този, който има асоцииран ***document type declaration (DTD)*** и документа спазва условията дефинирани там
- ***DTD*** може да бъде както външен маркъп (т.е. отделен файл, който се указва в декларацията на документа), така и част от документа (вътрешен ***DTD***)

ПРИМЕР
XML документ



СИМВОЛИ И МАРКЪП

Текстът се състои от поредица от символи и *маркъп*.

Маркъп приема формата на:
- отварящ таг, затварящ таг, таг за празен елемент
- код за символ
- коментар
- декларация за тип на документа, ***XML*** декларации
- функции за обработка
- текстови декларации
- всякакви празни символи, които са на най – горно ниво на документната същност (т.е. извън документната същност и извън всякакъв друг маркъп)

СИМВОЛИ И МАРКЪП

•Текстът, който не е маркъп представлява символните данни на един документ.
•Като съдържание на елементи - символни данни са всяка поредица от символни низове, които не съдържат маркъп разделител

ЗАПАЗЕНИ СИМВОЛИ

- Знаците & и < са запазени знаци и не трябва да се появяват в оригиналната си форма освен ако не са част от маркъп разделители.
- Ако тези знаци трябва да бъдат използвани като символни данни то те трябва да бъдат кодирани със съответните им числово-текстови кодове (&amp = '&', &lt = '<', &gt='>')
- Символите ' и " се кодират с &apos и &quot

КОМЕНТАРИ

- Коментари могат да се слагат навсякъде в един документ извън други маркъпи
- Коментарите не са част от символните данни на документа
- XML процесорът може да има функционалност, за достигането до съдържанието на коментарите, но това не е задължително

Коментар е всичко, което се намира между символните поредици <!-- и -->

```
<!-- това е коментар -->

<Element <!-- fsdfs --> id="0"> - неправилно!!!
```

ИНСТРУКЦИИ ЗА ОБРАБОТКА

- Инструкциите за обработка ***(Processing instructions – PI)*** позволяват документите да съдържат инструкции за приложенията
- Инструкциите за обработка не са част от символните данни на документа
- Инструкциите за обработка започват с *цел* ***(PITarget),*** която се използва да се укаже към кое приложение е насочена съответната инструкция
- *Целите* „xml“ и „ XML“ са запазени за стандартизация

```
<?Application type="Param"?>
```

CDATA секции

- ***CDATA*** секции могат да бъдат разположени навсякъде, където има символни данни.
- Използват се за добавяне на текст, който иначе би бил разпознат като маркъп
- В такава секция може да се използват символите „&“, „<“ без да е нужно да се кодират
- ***CDATA*** секции не могат да се влагат
- ***CDATA*** секция започва с „![CDATA[„ и завършва с „]]“

```
<![CDATA[<greeting>Hello, world!</greeting>]]>
```

ДЕКЛАРАЦИЯ ЗА ТИП НА ДОКУМЕНТА

XML документ, трябва да започва с декларация, където е указано коя версия на спецификацията използва.

Следващият пример е ***XML*** документ, който използва версия 1.1 на спецификацията. Той е добре оформен, но не валиден.

```
<?xml version="1.1"?>
<greeting>Hello, world!</greeting>
```

Следващият пример е ***XML*** документ, който използва версия 1.0 на спецификацията.

```
<greeting>Hello, world!</greeting>
```

ДЕКЛАРАЦИЯ ЗА ТИП НА ДОКУМЕНТА

Декларацията за типа на документа трябва да е преди първия елемент в документа

- Декларацията за тип на документа съдържа маркъп декларации, които предоставят граматика за клас от документи.
- Тази граматика се нарича ***document type declaration (DTD)***.
- Такава декларация може да бъде както външна, вътрешна така и смесена (вътрешна и външна)

Маркъп декларация е декларация за типа на елементите, декларация на списъка с атрибутите, декларация на същност

ПРИМЕР

```
<?xml version="1.1"?>
<!DOCTYPE greeting SYSTEM "hello.dtd">
<greeting>Hello, world!</greeting>
```

```
<?xml version="1.1" encoding="UTF-8" ?>
<!DOCTYPE greeting [
<!ELEMENT greeting (#PCDATA)>
]>
<greeting>Hello, world!</greeting>
```

САМОСТОЯТЕЛЕН ДОКУМЕНТ

Един ***XML*** документ може да бъде както самостоятелен, така и да съдържа препратки към външни същности.
Декларацията ***standalone*** указва дали съществуват декларации, които са външни за документа или не

```
<?xml version="1.1" standalone='yes'?>
```

Един ***XML*** документ може да бъде както самостоятелен, така и да се съдържа препратки към външни същности.
Ако ***standalone*** има стойност ***yes*** то този документ няма външни маркъп декларации. Ако стойността е ***no*** то този документ има или може да има външни маркъп декларации.

ОТВАРЯЩ ТАГ

- Един ***XML*** документ съдържа един или повече елемента.
- Един елемент е затворен между отварящ таг и затварящ таг
- Ако един елемент е празен, вместо отварящ таг и затварящ таг може да се напише един таг – таг за празен елемент
- Всеки елемент има тип, определен от името му
- Всеки елемент може да има атрибути
- Името на един затварящ таг трябва да съвпада с името на отварящия таг

```
<име> съдържание </име>
```

ОТВАРЯЩ ТАГ

- Всеки елемент, който не е празен, започва с отварящ таг
- Елементите могат да имат атрибути
- Атрибутите имат име и стойност
- Стойността е всичко, което се намира между ' ' или “ “
- Редът на атрибутите няма значение

```
<termdef id="dt-dog" term="dog">
```

ЗАТВАРЯЩ ТАГ

- Краят на всеки елемент, който има отварящ таг трябва да завършва със затварящ таг
- Името в затварящия таг трябва да е като това в отварящия таг
- Не могат да се добавят атрибути в затварящ таг

```
</termdef>
```

СЪДЪРЖАНИЕ НА ЕЛЕМЕНТ

Текстът, който се съдържа между отварящ и затварящ таг се нарича съдържание на елемента

```
<Name> Nenko Tabakov </Name>
```

ПРАЗЕН ЕЛЕМЕНТ

- Ако един елемент няма съдържание, той се нарича празен елемент
- Един празен елемент може да се запише с кратка форма – таг за празен елемент

```
<termdef> </termdef>

<termdef/>
```

DTD (Document Type Declaration)

•Един документ е валиден ако той изпълнява правилата описани в ***Document Type Declaration***
•***DTD*** може да бъде както част от ***XML*** документа, така и външен файл

С ***DTD*** може да се определи:
•типа на съдържанието на даден елемент
•броя на атрибутите да даден елемент, както и стойности по подразбиране
•броя на вложените елемент, ако има такива
•кои елементи могат да се влагат

DTD (Document Type Declaration)

DTD като част от ***XML*** документа

```
<!DOCTYPE PhoneBook [>
<!ELEMENT Element (Name, Phone, Email, Address)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Phone (#PCDATA)>
<!ELEMENT Address (#PCDATA)>
<!ELEMENT Email (#PCDATA)>

<!ATTLIST Element id CDATA "0">
]>
```

DTD (Document Type Declaration)

DTD като външен файл

```
<!DOCTYPE PhoneBook SYSTEM "example.dtd">

example.dtd:
<!ELEMENT Element (Name, Phone, Email, Address)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Phone (#PCDATA)>
<!ELEMENT Address (#PCDATA)>
<!ELEMENT Email (#PCDATA)>

<!ATTLIST Element id CDATA "0">
```

DTD (Document Type Declaration)

тип_на_елемента – може да бъде:

- ***CDATA***
- ***PCDATA***
- ***EMPTY***
- ***ANY***

```
<!ELEMENT име_на_елемент (тип_на_елемента)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Name EMPTY>
<!ELEMENT Name ANY>
```

DTD (Document Type Declaration)

Броят на вложените елементи може да бъде
- 1 – име_на_елемента
- 1 или повече – име_на_елемента+
- 0 или повече - име_на_елемента*
- 0 или 1 - име_на_елемента?

```
<!ELEMENT име_на_елемент (елемент1, елемент2, ...)>
<!ELEMENT Element (Name)>
<!ELEMENT Element (Name+)>
<!ELEMENT Element (Name*)>
<!ELEMENT Element (Name, Phone, Email, Address)>
```

DTD (Document Type Declaration)

Тип на стойността може да бъде:

- ***CDATA*** - текст
- *стойност1* | *стойност2* | ... - стойността е някоя от изброените
- ***ID*** – уникално име (трябва да започва с буква)

```
<!ATTLIST име_на_елемент име_на_атрибут тип_на_стойността
"стойност_по_подразбиране”>
<!ATTLIST Element id CDATA "0">
```

DTD (Document Type Declaration)

За стойност_по_подразбиране може да има една от следните думи:

- #REQUIRED – атрибутът е задължителен, но няма стойност по подразбиране
- #IMPLIED – атрибутът не е задължителен и няма стойност по подразбиране
- #FIXED – атрибутът има фиксирана стойност

```
<!ATTLIST Element id CDATA #REQUIRED>
<!ATTLIST Element id CDATA #IMPLIED>
<!ATTLIST Element id CDATA #FIXED „0“>
```