Przetwarzanie dokumentów XML i zaawansowane techniki WWW

"Wprowadzenie do tworzenia dokumentów XML"

(Zajęcia 02 - 07.03.2016 r.)

1) Czym jest XML (eXtensible Markup Language)

- przeznaczony do opisu struktury danych i ich powiązań z innymi danymi,
- w przeciwieństwie do HTML który opisuje sposób prezentacji danych, **XML je przechowuje**,
- XML jest zaprojektowany, aby przenosić dane, a nie format,
- jest rekomendowany przez W3C jako standard wymiany danych,
- jest przechowywany w zwykłych plikach tekstowych, które mają określoną strukturę,
- mogą być w łatwy sposób udostępniane pomiędzy komputerami

2) Przykładowe znane języki znaczników:

- HTML, LaTeX, RTF, SVG,
- obok samego tekstu (danych) ją specjalne znaczniki w interpretowane przez dany parser,
- przykład z html: `<b>` tekst pogrubione `<b>,`
- przykład Latex:

```
\begin{equation}
      s = \int L dq
\end{equation}
```

- w odróżnieniu od HTML znaczniki definiowane są przez użytkownika, a ich nazewnictwo odnosi się bezpośrednio do typu i znaczenia przechowywanie danych,
- w XML wszystko zależy od autora,
- natomiast istnieje zestaw fundamentalnych reguł dotycznących formatowania pliku XML, jego używania i manipulowanie,

3) Zalety / wady i zastosowanie:

ZALETY:

- kod w zwykłych plikach tekstowych co pozwala na łatwy dostęp oraz czytelną i jasną interpretację,
- obsługa UNICODE,
- otwarty standard - niepotrzebne są żadne licencje,
- niezależność od platformy - łatwa przenoszlność,
- "ścisły format" - który pozwala na łatwą, szybką i wydajną interpretację:

WADY:

- czasem pliki mogą być za duże, oraz ze zbyt dyżym stopniem zagnieżdzenia elementów,
- zwykle nie da się zastosować plików XML dla przechowywania danych "chaotycznych"
- wydajność niektórych implementacji XML może być niewystarczająca,

ZASTOSOWANIE:

- przechowywanie danych strukturalnych, pliki konfiguracyjne, książki adresowe, małe magazyny danych jako alternatywa dla baz danych np SQL,
- wymiana danych pomiędzy aplikacjami z różnych platform systemowych,
- udostępnianie danych,
- tworzenie nowych języków tzw. metajęzyków,

4) Terminologia:

- **ZNACZNIK**: to symboliczny identyfikator otoczony znakami

  `<znacznik> lub </znacznik>`

  przykład z HTML: `<b> <i>`.
- **ELEMENT (WĘZEŁ - NODE)** - posiada znacznik końcowy i początkowy

  `<b> Tekst lub inne znaczniki </b>`
- węzły moga w sobie zawierać inne węzły (zagnieżdzanie),
- moga zwierać zwykły tekst
- moga być puste,
- Węzły które nie zawierają pomiędzy znacznikiem otwierającym i zamykającym tekstu lub innych elementów nazywamy węzłami pustymi i czasem oznaczamy

  `<znacznik/>`

  w XHTML np. `<br/> <hr/>`
- **WĘZEŁ GŁÓWNY (ROOT NODE)** najwyższy węzeł w hierarchi zawierający wszystkie pozostałe węzły,
- **ATRYBUTY** - elementy moga psiadać specjalne atrybuty, które sa swoistymi dodatkowymi identyfikatorami, dołączonymi do znaczników w w postaci

  `nazwa-atrybutu='wartosc'`

  przykład z HTML:

```
<div class="box">
  <a id="link" href="http://www.uj.edu.pl">Link </a>
</div>
```

- elementy w XML tworzą strukturę, w której elementy zawierające inne elementy są dla nich przodkami, (lub węzłami nadrzędnymi) - tzw. ***parent nodes***, natomiast elementy zawarte w nich elementami podrzędnymi - tzw. ***child nodes***.

5) PIERWSZY XML:

Przed przystąpieniem do projektowania struktury pliku XML, programista musi odpowiedzieć na następujące podstawowe pytania:

1. **Czy dane które chcemy umieścić w pliku XML wogóle się do tego nadają - mają strukturę hierarchiczną?**
2. **Czy dane mają jasno określone własności?**
3. **Czy te własności mogą być określone za pomocą atrybutów i elementów podrzędnych?**

```
<?xml version="1.0" encoding="UTF-8"?>
<osoba>
      <imie>Jan</imie>
      <nazwisko>Kowalski</nazwisko>
      <pesel>83010198741</pesel>
</osoba>
```

prolog

root-node (element główny)

Prolog - powinien być ale nie zawsze jest konieczny!!!

`?xml` - deklaracja typu dokumentu
`version` - wersja standardu: 1.0 - 11,2008, 1,1 - 08,2006
`encoding` - kodowanie może też być iso-8859-2

w prologu moga znaleźć się też inne elementy np. informacja o styłach lub dołączonym pliku DTD (Document Type Definition).

Podstawowe zasady tworzenia znaczników:

- nazwy elementów mogą być dowolnym ciągiem znaków rozpoczynającym się do litery lub podkreślenia, ale nie moga rozpoczynać się od cyfry.
- jak w każdym meta języku możemy dodawać komentarze:

```
<!-- komentarz ->
```

ale nie moga się pojawić przed deklaracją dokumentu XML,
- nazwy atrybutów tworzymy podobnie jak nazwy znaczników rozpoczynając od litery lub podkreslenia, ale nigdy od cyfry.

Ze wzlędu na charakter meta-języka XML pewne znaki specjalne są zastrzeżone dla składni języka. Gdy chcemy użyć takiego znaku specjalnego powinnismy stosowac tzw. encje znakowe:

```
<  - &lt;
>  - &gt;
&  - &amp;
'  - &apos;
"  - &quot;
```

6) Przestrzenie nazw

Pozwalają na rozwiązanie problemu identyfikacji danych znaczników - np. wymiana danych między dwoma aplikacjami:

```
<hc:Dane xmlns:hc="test">
      <hc:Pacjent>
      <hc:/Pacjent>
<hc:/Dane>
```

`hc:` - jest to prefix,
`test` - nazwa przestrzeni określonych "nazw".

Nazwy przestrzeni mogą stanowić adresy URI:

```
<h:table xmlns:h="www.w3.org/TR/html/h">
<f:table xmlns:f="www.w3.org/TR/html/f">
```

6) Stylizacja dokumentów XML do wyświetlania za pomocą kaskadowych arkuszy stylu (CSS)

Do formatowanie i wyświetlania danych z plików XML można użyć kaskadowych arkuszy stylu (CSS). W tym przypadku podobnie jak w HTML, uzywamy selektorów które bezpośrednio wskazują na konkretne elementy pliku XML (np. selektrami mogą być nazwy znaczników). Dla każdego selektora tworzymy regułę określającą sposób prezentacji danych w przeglądarce. .

Przykład selektora z regułą:

```
KOMIS {
            display: block;
            background-colr: #AAAAA;
            color: #FFFFFF;
```

```
	}
```

oraz pliku dla którego ta reguła będzie zastosowana:

```
<?xml version=”1.0” encoding=”utf-8”?>
<?xml-stylesheet type=”text/css” href=”styl.css”?>
<KOMIS>
	<Samochod>
		<marka>BMW</marka>
	</Samochod>
</KOMIS>
```

Można, poza selektorami tzw. znacznikowymi stosować: identyfikatory oraz klasy podobnie jak ma to miejsce w języku HTML. W przypadku chęci nadania odpowiedniego stylu konkretnemu znacznikowi dodajemy atrubut o nazwie “id” i przypisujemy mu nazwe identyfikującą. Dany identyfikator może występować tylko raz w pojedyńczym dokumencie XML. Przykład:

```
KOMIS {
		display: block;
		background-color: #CCCCCC;
		color: #000000;
	}

#bmw {
	 color:green;
}

#mazda {
	 color:red;
}


<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="styl.css"?>
<KOMIS>
	<Samochod>
		<marka id="bmw">BMW</marka>
			<marka id="mazda">Mazda</marka>
			<marka class="ww">Audi</marka>
	</Samochod>
</KOMIS>
```

W przypadku selektorów tzw. “klas” sprawa jest nieco trudniejsza niż w przypadku języka HTML. W meta-języku XML, aby nadać styl grupie znaczników identyfikowanych tą samą nazwą “klasy” należy użyć pełnego zdefiniowania dla selektora klasy. Rozbudowując poprzedni przykład:

```
KOMIS {
		display: block;
		background-color: #CCCCCC;
		color: #000000;
	}
```

```
#bmw {
      color:green;
}

#mazda {
      color:red;
}

marka[class=all] {
     color:yellow;
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="styl.css"?>
<KOMIS>
      <Samochod>
                  <marka class="all">Honda</marka>
                  <marka id="bmw">BMW</marka>
                  <marka id="mazda">Mazda</marka>
                  <marka class="all">Audi</marka>
      </Samochod>
</KOMIS>
```

Zadanie

Proszę przygotować plik XML z danymi tabelarycznymi, oraz plik z regułami CSS który wyświetli dane w przeglądarce w postaci tabeli.

Do tworzenia plików XML i styli CSS wystarczy podstawowy edytor tekstu np. Notepad w systemie Windows lub Gedit w systemi Linux. Do oglądania efektów pracy wystarczy dowolna przeglądarka internetowa.