MAGYARI MÁRK
CWLKE4
PTI
2021.01.12.

1)

SAX

- A parser szekvenciálisan dolgozza fel az xml dokumentumot.
- Nagy adatbázisnál is jól működik (gyors)
- startElement, endElement

DOM

- Fa-szerkezetet készít és beolvassa a memóriába
- Nagy adatbázisnál lassú
- Gyerek-szülő kapcsolat

2)

Az xml dokumentumban egy fa struktúra alapján elérési útvonalat biztosít.

Adattípusai:

- elem csomópont
- text csomópont
- attribútum csomópont
- névtér csomópont

Tengely típusai:

- self
- child
- ascendant
- ancestor
- sibling

Operátorok:

+ összeadás
- kivonás
* szorzás
| egyesítés
mod modulo
div osztás

2021.01.12.

3.) DOM - platform és nyelvfüggetlen

Az xml-ről egy fa-struktúrát készít, mint objektumokat kezel. Elementeket node-okként reprezentál pl.: HTML-nél p, h1, stb. gyerek-szülő kapcsolatok. A DOM lehetővé teszi a csomópontok készítését, törlését, megtudja változtatni a dokumentum tartalmát, szerkezetét. Nem csak egy xml esetén érdemes.

Metódusai:

- appendChild()
- removeChild()
- getElementById()
- getElementsByTagName()
- createElement()

4.) Az xml schema az xml dokumentumra kiterjedő struktúrát és szabályokat hoz létre. Meghatározza, hogy milyen sorrend lehet, milyen típusok, elemek, hányszor fordulhatnak elő.

DTD előnyei:

- egyszerű használhatóság, szerkezet
- ~~támogatja a névtereket~~

Hátrányai:

- az előfordulást nem lehet szabályozni
- ~~támogatja a stb~~
- nem támogatja a névtereket

5) 2021.01.12.

Az XSL az xml dokumentumból állít elő egy másik dokumentumot, pl: HTML, pdf, stb.

Kimenet: annilyen szintaktikát kell írnunk pl: html-re

```
<html>
  <body>
    <p>
    </p>
  </body>
</html>
```

Outputképzés:

```
<xsl:for-each-group select="." group-by="." >
  ...
</xsl:for-each-group>
```

renderes

```
<xsl:sort >
  ...
</xsl:sort>
```