

XML

Sisältää XML datan käsittelyyn tarvittavia funktioita. Perustuu TinyXML2 -kirjastoon.

Xml.parseAsLuaDOM (strXml)

Xml.loadAsLuaDOM (strFile)

strXml Merkkijono joka sisältää XML dataa

strFile Tiedosto josta XML data ladataan

Parsii XML annetun datan, ja palauttaa sen lua taulukkona.

Tuloksena syntyvä lua taulukko seuraa seuraavaa DOM -mallia mukailevaa rakennetta:

-- Esimerkki XML data:

```
xmlData = [[
<kirjat>
  <kirja muoto="kovakantinen">
    <kirjailija>Pekka Meikäläinen</kirjailija>
    <vuosi>1975</vuosi>
  </kirja>
  <kirja muoto="pokkari">
    <kirjailija>Matti Meikäläinen</kirjailija>
    <vuosi>1974</vuosi>
  </kirja>
</kirjat> ]]
```

-- Funktiokutsu:

```
local d = Xml.parseAsLuaDOM (xmlString)
```

>> tulos

```
d = {
  _name = "kirjat",
  _attr = {},
  _text = "",
  kirja = {
    _name = "kirja",
```

```
_attr = {muoto="pokkari"},
_text = "",
kirjailija={
  _name = "kirjailija",
  _attr = {},
  _text = "Matti Meikäläinen"
},
vuosi={
  _name = "vuosi",
  _attr = {},
  _text = "1974"
}
[1] = {
  _name = "kirjailija",
  _attr = {},
  _text = "Matti Meikäläinen"
}
[2] = {
  _name = "vuosi",
  _attr = {},
  _text = "1974"
}
},
[1] = {
  _name = "kirja",
  _attr = {muoto="kovakantinen"},
  _text = "",
  kirjailija={
    _name = "kirjailija",
    _attr = {},
    _text = "Pekka Meikäläinen"
  },
  vuosi={
    _name = "vuosi",
    _attr = {},
    _text = "1975"
  },
  [1] = {
    _name = "kirjailija",
    _attr = {},
```

```

    _text = " Pekka Meikäläinen"
  },
  [2] = {
    _name = "vuosi",
    _attr = {},
    _text = "1975"
  }
},
[2] = {
  _name = "kirja",
  _attr = {muoto="pokkari"},
  _text = "",
  kirjailija={
    _name = "kirjailija",
    _attr = {},
    _text = "Matti Meikäläinen"
  },
  vuosi={
    _name = "vuosi",
    _attr = {},
    _text = "1974"
  },
  [1] = {
    _name = "kirjailija",
    _attr = {},
    _text = "Matti Meikäläinen"
  },
  [2] = {
    _name = "vuosi",
    _attr = {},
    _text = "1974"
  }
}
}

```

Yllä olevasta esimerkistä käy ilmi, että tuloksena yksinkertaisesta XML datasta on melko monimutkainen lua taulukko. Syitä siihen on monia; XML dokumentin perusyksikkö on n.s. node, eli solmu. XML dokumentti koostuu määritelmän mukaan yhdestä juuri elementistä – esimerkissä nimeltään kirjat – joka voi puumaisesti sisältää minkä tahansa määrän alisolmuja. Koska nämä

solmut voivat olla nimeltään identtisiä, täytyy XML dokumenttia parsiessa säilyttää paitsi solmun nimi, myös niiden järjestys. Koska lua taulukoiden tapauksessa on usein näppärämpää viitata taulukon alkioihin nimellä kuin indeksillä, tässä tapauksessa molemmat.; Lapsi solmut lisätään lua- taulukkoon paitsi juoksevasti numeroiden, myös solmu nimellään, jolloin viimeinen lisätty solmu jää voimaan. Esimerkin tapauksessa "kirjat.kirja" osoituksen takaa löytyy Matti Meikäläisen kirjoittama kirja.

Tässä mallissa säilytetään siis kaikki XML dokumenttiin sisältyvä tieto.

Xml.loadAsTable (strXml)

Xml.parseAsTable (strFile)

strXml Merkkijono joka sisältää XML dataa

strFile Tiedosto josta XML data ladataan

Lataa annetun XML datan ja palauttaa tuloksen yksinkertaisena taulukkona. Tuloksena syntyvä lua -taulukko on yksinkertaisempi kuin DOM mallia seuraava, mutta se ei esimerkiksi ymmärrä solmujen attribuutteja, tai saman nimisiä lapsisolmuja.

Esimerkki:

-- Esimerkki XML data:

```
xmlData = [[
<kirjat>
  <kirja muoto="kovakantinen">
    <kirjailija>Pekka Meikäläinen</kirjailija>
    <vuosi>1975</vuosi>
  </kirja>
  <kirja muoto="pokkari">
    <kirjailija>Matti Meikäläinen</kirjailija>
    <vuosi>1974</vuosi>
  </kirja>

</kirjat> ]]

-- Funktiokutsu:
local d = Xml.parseAsLuaDOM (xmlString)

>> tulos
d = {
  kirjat={
    kirja={
      kirjailija="Matti Meikäläinen",
```

```
        vuosi="1974"
    }
}
}
```

Xml.serialize (o, r)

o Lua objekti joka muutetaan XML muotoon.

r Dokumentin juurisolmun nimi

Muuttaa hyvin suoraviiveisesti lua-aulukon XML muotoon.

Huomaa, että lua-aulukon numeraaliset indeksit eivät käänny XML muotoon oikein. Funktio muuttaa ne <1>, <2>, .. jne nimisiksi solmuiksi, mutta nämä eivät ole sallittua XML:ää.

Esimerkki:

-- Esimerkki XML data:

```
data = {
    description = "testi data",
    pv = 90.5,
    polarity = "normal"
}

local xml = Xml.serialize (data, "testNode")

>> tulos
xml = [[
<testNode>
    <description>testi data</description>
    <pv>90.5</pv>
    <polarity>normal</polarity>
</testNode>
]]
```

Revision #3

Created 25 May 2022 10:50:57 by Severi Hiltunen

Updated 10 May 2023 10:25:34 by Severi Hiltunen