Sirhán Bálint

Digitális tartalmak három-dimenzióban, webes környezetben történő megjelenítése: *CSS3D és X3D*

A XXI. században az informatika szinte az élet minden területén megjelent, aminek eredményeképpen hatalmas mennyiségű digitális adattartalom született. Ahhoz, hogy egyes tartalmakat a weben keresztül is hozzáférhetővé lehessen tenni − ingyenesen vagy szolgáltatás keretében − többféle szabvány, technológia került már kidolgozásra. A hálózaton keresztül történő dokumentum megosztásnak − ami például egy könyvtár esetében a helyismereti gyűjtemény közzététele, a digitalizálást követően − egyik hatékony módja a tartalom webes környezetbe történő ágyazása, mondjuk egy galéria. Ez önmagában ma már nem újdonság, sőt alapelvárás, hogy a rendelkezésre álló digitális gyűjteményt − természetesen a szerzői jogok figyelembe vételével − a felhasználó számára megfelelő módon megtekinthetővé, böngészhetővé kell tenni. Ahhoz, hogy ez minél látványosabb, interaktívabb és élménydúsabb legyen, érdemes 3D technológiát alkalmazni.

Tárgyszavak: *digitális archívum, digitális dokumentum, weblap; háromdimenziós dokumentum*

Bevezetés

Manapság a 3D technológia nem csak a szórakozás egyik eszköze, hiszen a tudomány több területén is használják már, például az építészetnél. Utóbbi esetében még csak a tervező asztalon létező ingatlant, az épületmodellezésnek köszönhetően, már előre, akár teljes egészében megtekinthetjük, sőt be is járhatjuk. Ehhez nincs másra szükség, mint a megfelelő szoftver (pl. ArchiCAD) kiválasztására és kezdődhet is a tervezői munka. Azonban most nem a különböző 3D-s tervezőszoftverekről lesz szó, hanem azokat a technológiákat mutatom be, amelyek lehetővé teszik, hogy a létrejött digitális tartalmakat három dimenzióban, webes környezetben megjelenítsük. Alapvető kérdés, hogy a vizualizáció eléréséhez egy sima böngészőben (pl. Google Chrome) szükséges-e valamilyen plugin használata. Ez főleg a korábbi évek problematikája volt, ma már léteznek olyan szabványok/specifikációk, amelyek nem igényelnek semmilyen külső bővítményt. A cikkben két fő fejlesztési lehetőséget − a **CSS3D**, illetve az **X3D** technológia − részletezek, amelyek kellően hatékonyak ahhoz, hogy valós idejű, megfelelő minőségű, élvezhető háromdimenziós tartalommegjelenítést biztosítsanak.

CSS3 3D Transforms – a webfejlesztők, webdesignerek gyöngyszeme

1. ábra **Cascading Style Sheets 3D**

A CSS stílusleíró nyelv *(1. ábra)* gyakorlatilag a honlapok külleméért felel, manapság a webre történő fejlesztéskor elkerülhetetlen ennek a használata. A kódolás ebben az esetben nem annyira bonyolult, azonban a háromdimenziós élmény eléréséhez a CSS Level 3-at kell alkalmaznunk, amely lehetővé teszi a HTML elemek 3D-s transzformációját. Azért is érdemes a CSS-t használni a 3D-s élmény eléréshez, mert nincs kifejezetten szükség a GPU hardveres gyorsítására. Ez a gyakorlatban azt jelenti, hogy a kisebb könyvtárakban

vagy múzeumokban használt számítógépeken is probléma nélkül megtekinthetők ezek a tartalmak. A gyakorlatban a legegyszerűbben, a CSS3D-t úgy lehet szemléltetni, hogy leképezünk egy háromdimenziós kockát, amihez szükséges egy HTML (v5) és egy CSS (v3) állomány. Előbbi esetében tagek segítésével deklaráljuk a főbb paramétereket, az-az, magát a tartalmat, utóbbival pedig a megjelenési stílust határozzuk meg.

a) HTML állomány kódolása

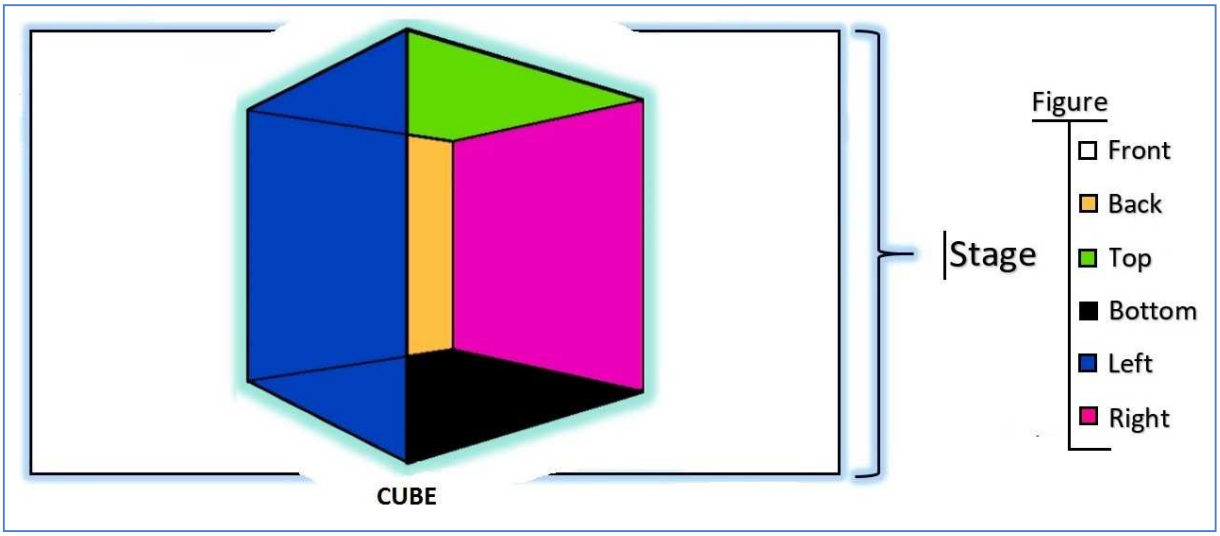
```
<div class="stage">
  <div class="cube">
    <figure class="front"></figure>
    <figure class="back"></figure>
    <figure class="top"></figure>
    <figure
class="bottom"></figure>
    <figure class="left"></figure>
    <figure class="right"></figure>
  </div>
</div
```

ML struktúráját tekintve két fő részt kell megadni, az egyik `<div class="stage">` elem, ami tulajdonképpen a „színpadot” jelenti, ide fog minden kerülni. A második magának a megjelenített objektumnak (kocka) a definiálása, amit a `<div class="cube">` tag-gel tehetünk meg. Jelen példában a `<figure>` elemmel adjuk meg a kocka 6 oldalát. A megjelenítés szempontjából, az első oldalt, amit a kockából látunk, a `<div class="front">` elemmel definiáljuk. Ezt követően adjuk meg a bal/jobb és alsó/felső, illetve hátsó oldalra vonatkozó tageket is *(2. ábra).*

ML struktúráját tekintve két fő részt kell megadni, az egyik `<div class="stage">` elem, ami tulajdonképpen a „színpadot” jelenti, ide fog minden kerülni. A második magának a megjelenített objektumnak (kocka) a definiálása, amit a `<div class="cube">` tag-gel tehetünk meg. Jelen példában a `<figure>` elemmel adjuk meg a kocka 6 oldalát. A megjelenítés szempontjából, az első oldalt, amit a kockából látunk, a `<div class="front">` elemmel definiáljuk. Ezt követően adjuk meg a bal/jobb és alsó/felső, illetve hátsó oldalra vonatkozó tageket is *(2. ábra).*

Selector 1.

```
stage {
  width: 300px; height: 250px;
  -webkit-perspective: 1600px;
  -webkit-perspective-origin: 50% -
240px;
}
```



2. ábra **A kocka elemeinek definiálása**

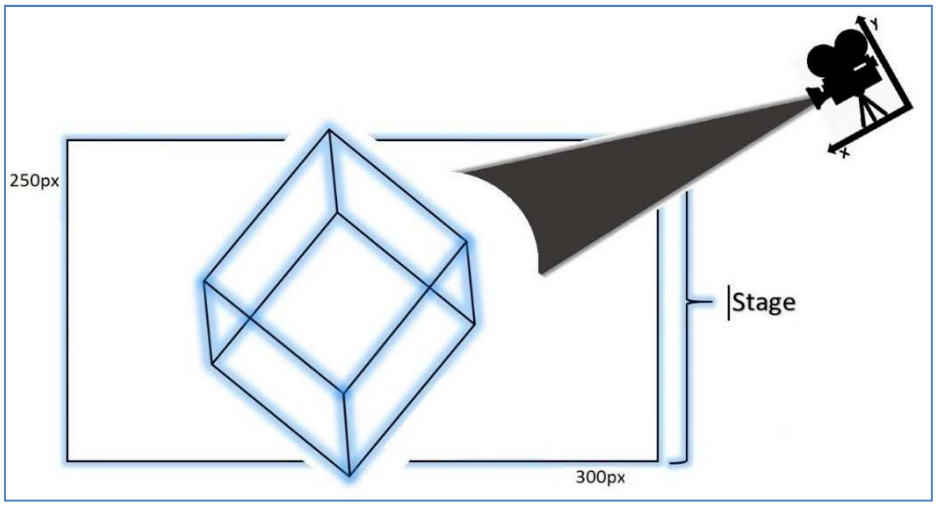
Ezt követően a „perspective” érték beállításával elérjük a 3D hatást, ami annál kifinomultabb lesz, minél nagyobb számot írunk. Ne feledjük a „perspective-origin” értéket sem, ami a „virtuális kamerának” az elhelyezkedése az X és Y tengely mentén. Gyakorlatilag ebből a szögből fogjuk látni a színpadra helyezett objektumokat *(3. ábra).*

Második lépésként ki kell rajzoltatni az alakzatot, a „figure” szelektor utasítással. Mivel korábban a HTML struktúrának megfelelően 6 db *figure* elemet definiáltunk, így összesen hat négyzetet rajzolunk: (ezek a kocka egyes oldalai).

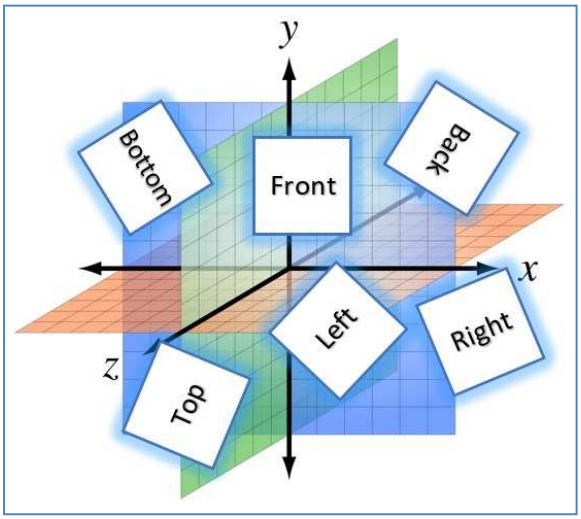
Selector 2.

```
figure {
  display: block; position: absolute;
  width: 300px; height: 300px;
  background-color: #60c2ef;
}
```

A harmadik lépésként meg kell adni, hogy a 3D-s terünkben hol helyezkedjenek el az egyes négyzetek, összeállva így egyetlen objektummá-kockává *(4. ábra).*



3. ábra **A színpad és a „virtuális kamera” szemléltetése**



4. ábra **A négyzetek elhelyezése az X-Y-Z tengely mentén**

Ehhez az egyes síkidomokat térben transzformálni kell, tehát forgatni, eltolni és skálázni az X, Y és Z tengely mentén:

Selector 3.

```
.front {
-webkit-transform:
translateZ(150px);
}
```

Selector 4.

```
.back {
-webkit-transform: rotateY(180deg)
translateZ(150px);
}
```

Selector 5.

```
.top {
-webkit-transform: rotateX(90deg)
translateZ(150px);
}
```

Selector 6.

```
.bottom {
-webkit-transform: rotateX(-90deg)
translateZ(150px);
}
```

Selector 7.

```
.left {
-webkit-transform: rotateY(-90deg)
translateZ(150px);
}
```

Selector 8.

```
.right {
-webkit-transform: rotateY(90deg)
translateZ(150px);
}
```

Ha ezzel megvagyunk, negyedik lépésként a már kész kockát mint egész objektumot is el kell forgatunk 3D-ben, azaz 360 fokban. Ehhez meg kell adni a „`.cube`” és a „`keyframes rotate`” transzformációs utasításokat:

Selector 9.

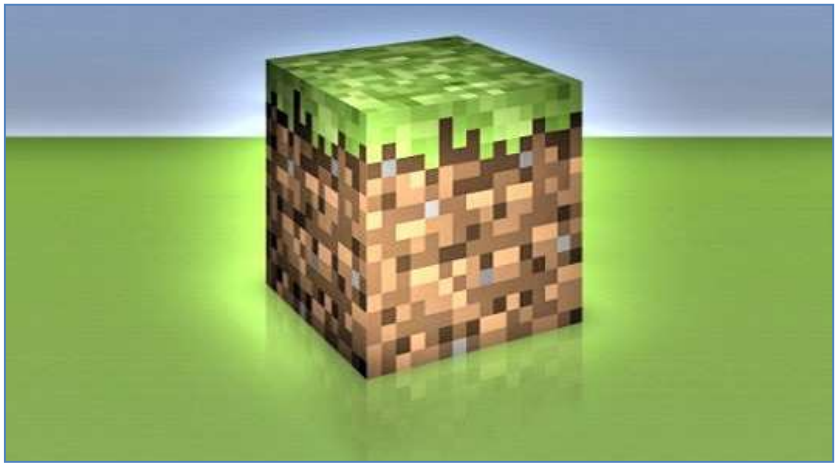
```
.cube {
-webkit-transform-style: preserve-
3d;
}
```

Selector 10.

```
@-webkit-keyframes rotate {
  0% { -webkit-transform:
rotateY(0); }
  100% { -webkit-transform:
rotateY(360deg); }
}
```

Extra lehetőségként, kedvenc kockánkat textúrázni is tudjuk. Ez a gyakorlatban annyit tesz, hogy színt vagy valamilyen mintát adunk az adott objektumnak *(5. ábra)*.

Természetesen rengeteg még a CSS3D-ben rejlő lehetőség, például animációk, teljes virtuális terek létrehozása, a fentiek csak egy nagyon kis szelete az egésznek. Ami mindenképpen megvalósítható ezzel a technológiával, az nem más, mint az interaktivitás. Egy múzeum esetében, ha az adott kiállítás 3D-ben kerül bemutatásra, minden bizonnyal sokkal izgalmasabb és magával ragadóbb lesz a látogatók számára. Az egyes műtárgyak „3D digitalizálása” akár értékmegőrző szerepet is betölthet, egyúttal duplikálható is bizonyos szinten, mivel a létrejött modellek nyomtathatók. Ha egy közgyűjtemény teljes weboldala 3D-ben böngészhető, sokkal könnyebben hozzáférhetővé válik a sérült emberek számára is. Például a látássérülteknek, akik nehezebben olvasnak, a háromdimenzióban megjelenített szöveg valóban jobban érzékelhetők. Nyilván az efféle tartalommegjelenítés bármilyen közgyűjteményi intézmény számára bizonyos anyagi terhet jelenthet, ezért érdemes előtte felmérni, van-e igény az ilyen jellegű szolgáltatásra. [5. 6, 7.]



5. ábra **3D Kocka - textúrázva és háttérrel**

X3D (Extensible 3D Graphics)

6. ábra **A konzorcium emblémája**

A platformfüggetlen, kiterjeszthető 3D grafika tulajdonképpen a VRML (Virtual Reality Modelling Language) szkriptnyelv továbbfejlesztett változata, amely még hatékonyabb (gyorsabb) adatátvitelt tesz lehetővé a hálózati alkalmazások számára. Az X3D a Web3D Konzorcium grafikai munkacsoportja (Graphics Working Group) által kialakított ISO/IEC 19775 nyílt szabvány, amelyet a Nemzetközi Szabványügyi Szervezet még 2004-ben hagyott jóvá. A formátum XML alapú, a JavaScripthez hasonlóan nagyon jól implementálható a HTML5-be, egyúttal igyekszik minél jobban kihasználni a GPU adta lehetőségeket. A 3D-s modellek leképezéséhez az X3 DOM-ot (Document Object Model) használ, ami egy open-source javascript függvénykönyvtár és amely már támogatja az olyan VR (virtuális valóság) technológiákat is, mint az Oculus Rift vagy a Microsoft HoloLens. A legfrisseb 3.3-as verzió jóval hatékonyabban képes a multi-stage/texture renderelésre, ami annyit jelent, hogy még több vizuális effekt leképzésére van lehetőség (light/environment mapping). Az X3D szabványt nyugodtan nevezhetjük egy „3D-for-all" technológiának, amelynek alkalmazásával bármilyen háromdimenziós környezet, színtér, alakzat és objektum megjeleníthető egy sima webböngészőben (például Mozilla Firefox).

7. ábra **Az olaszországi Siena székesegyház X3D modellje**
(http://examples.x3dom.org/Demos/Siena/siena.html)

Nézzük meg, hogy a gyakorlatban hogyan is épül fel, milyen struktúrával rendelkezik maga az X3D, mit és hogyan kell kódolni:

1. rész

```
<?xml version="1.0" encoding="UTF 8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.3//EN"
"http://www.web3d.org/specifications/x3d-3.3.dtd">
<X3D profile='Immersive' version='3.3'
xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance'
xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d
-3.3.xsd'>
```

Alapesetben mindig definiálni kell a !DOCTYPE X3D tag segítésével a HTML dokumentum típusát. A dtd (document type definition) fájlra történő hivatkozás is kötelező, hiszen ez tartalmazza az X3D szabvány validált követelményeit. Emellett a profile-ban meg kell adnunk az XML sémának (xsd) a hivatkozását és verziószámát, hogy egy érvényes XML fájlt kapjunk.

2. rész

```
<head>
  <meta content='HelloWorld.x3d' name='title'/>
  <meta content='Simple X3D example' name='description'/>
  <meta content='30 October 2000' name='created'/>
  <meta content='7 August 2010' name='modified'/>
  <meta content='Don Brutzman' name='creator'/>
  <meta content='http://www.web3D.org' name='reference'/>
  <meta
  content='http://www.web3d.org/x3d/content/examples/HelloWorld.x3d'
  name='identifier'/>
  <meta content='http://www.web3d.org/x3d/content/examples/
  HelloWorldTall.png' name='image'/>
  <meta content='http://www.web3d.org/x3d/content/examples/
  license.html' name='license'/>
</head>
```

A fejlécben a leíró(meta) adatokat kell megadni a meta content elemmel. Itt többek között meg kell határozni a címet (title), a létrehozót (creator), a létrehozás/módosítás dátumát (created/modified), az X3D fájl azonosítóját (identifier) és a megfelelő licenszre történő hivatkozást (license).

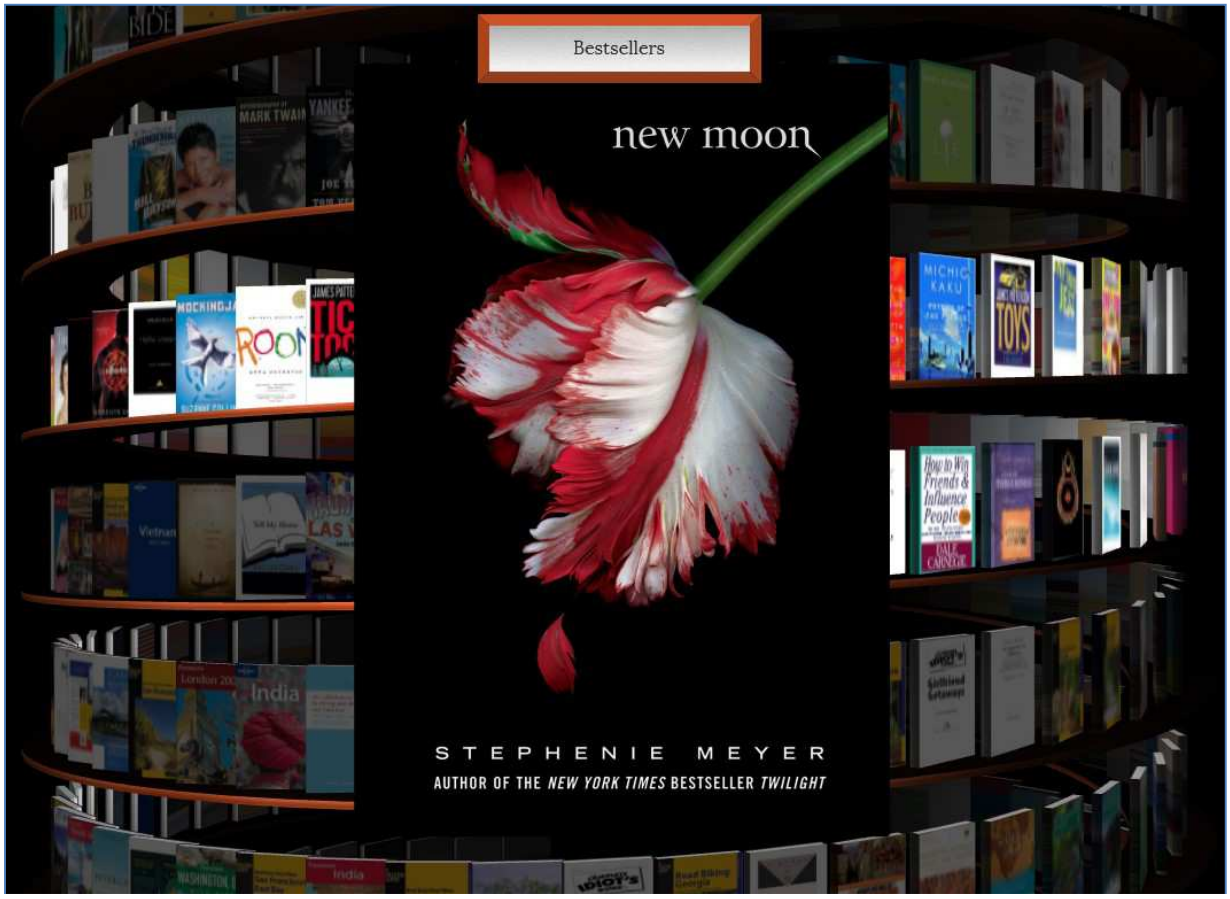
3. rész

```
<Scene>
  <Group>
   <Viewpoint centerOfRotation='0 1 0'
        description='Hello world!' position='0 1 7'/>
   <Transform rotation='0 1 0 3'>
    <Shape>
     <Appearance>
      <Material diffuseColor='0 0.5 1'/>
      <ImageTexture url='"earth topo.png"/>
     </Appearance>
     <IndexedFaceSet solid="false" coordIndex="0 1 3 2 -1 0 4 5 1 -1">
      <Coordinate point="0 0 0, 0 0 1, 1 0 0, 1 0 1, 0 1 0, 0 1 1"/>
     </IndexedFaceSet>
    </Shape>
   </Transform>
  </Group>
</Scene>
</X3D>
```

Az egyik legfontosabb rész az X3D kapcsán, a Scene elemek meghatározása, ami gyakorlatilag a háromdimenziós színtér. A Viewpoint elem beállítása azt a nézőpontot jelenti, ahonnan láthatók lesznek a színtérben megjelenített objektumok. Ami az igazán lényeges, hogy a Shape elemmel megadjuk az egyes objektumok attribútumait, például milyen alakzata legyen, kör vagy esetleg négyzet. Ez az X, Y és Z tengely mentén megadott koordinátákkal (Coordinate point elem) érhető el, a megfelelő értékekkel pedig „megrajzoljuk” az adott objektum formáját. Ezt követően lehetőség van textúrázásra is, ami az „ImageTexture” elemmel valósítható meg, így egy szép, élethű 3D-s modell hozható létre. [2, 4, 10]

Ha komplexebb 3D portált szeretnénk létrehozni, érdemesebb az X3D-t alkalmazni, mivel a webes környezetben való tartalommegjelenítésre leginkább ez a szabvány alkalmas. A kompatibilitás szempontjából is ez a jobb lehetőség, hiszen a 3D-s grafikai tervezőprogramok nagy részével képes együttműködni. A közismert, nyílt forráskódú Blender szoftverrel készített háromdimenziós modellek szintén exportálhatók X3D formátumban, így bármely weboldalba beágyazhatók. Természetesen a CSS3D is jó alternatíva lehet, érthetőbb és egyszerűbb a kódolása, egy könyvtár számára alkalmas lehet akár egy 3D-s könyvespolc létrehozására szép- vagy szakirodalomnak megfelelően *(8. ábra).*

Érdemes még megemlíteni a WebGL (Web-based Graphics Library) technológiát, amely teljes egészében webalapú. Ez a grafikus programkönyvtár is viszonylag könnyedén kódolható HTML5-höz és JavaScripthez. Egyik nagy előnye szintén a platformfüggetlensége, tehát bármilyen eszközön, operációs rendszeren és támogatott böngészőn megjeleníthetünk vele 3D-s tartalmakat. A WebGL hasonlóan az X3D-hez, hatékonyan képes együttműködni a GPU-val, de ez nem meglepő, mert az OpenGL ES grafikus API-n alapszik. Ezzel is kiváló, interaktív háromdimenziós élmény érthető el. [9]



8. ábra **3D-s könyvespolc (forrás: chromeexperiments.com)**

Aki ma a webre fejleszt plugin nélküli 3D-s alkalmazásokat, honlapokat, a felsorolt technológiák egyikét kell használja. Azonban a jövő, valószínűleg a ma még annyira nem elterjedt **WebVR**, azaz a böngészőbe ültethető virtuális valóság lesz *(9. ábra).* A WebVR alapulhat például az X3D szabványon is, a specifikáció annyit tesz hozzá − a már meglévő 3D-s technológiához −, hogy képes felismerni, ha egy VR szemüveg csatlakozik a rendszerhez. Ezen felül érzékeli, hogy a térben éppen merre nézünk, így annak megfelelően közvetíti az információt, bár ez függ az adott eszköztől is. Sajnos magyarországi viszonylatban a közgyűjteményi intézmények kevésbé tudják vagy akarják ezekben a technológiákban rejlő lehetőségeket kihasználni, főként gazdasági, illetve a szerzői jogi okok miatt. Külföldön viszont, főleg a múzeumok esetében, már megfigyelhető némi paradigmaváltás, például a British Museumnak − a Sketchfabbal történő együttműködésnek köszönhetően − már számtalan 3D-s modellje létezik.

9. ábra **Oculus Rift VR eszköz (forrás: oculus.com)**

A könyvtáraknál hazánkban bár elérhető néhány helyen virtuális kiállítás, azonban ezek csak 2D-s környezetben működnek. A cikkben írt specifikációk open-source alapúak, így bárki szabadon felhasználhatja őket. Nyilván a megfelelő szaktudás elengedhetetlen, azonban, ha van igény és elköte-

lezettség, érdemes a három dimenzióban rejlő lehetőségeket kihasználni.

Irodalom

1. BECKER, Bernd: To 3D or Not to 3D. In: Behavioral and Social Sciences Librarian, 2016. 35. sz., p. 83-86.
2. BRUTZMAN, Don − DALY, Leonard: X3D, Extensible 3D Graphics for Web Authors. – Amsterdam: Morgan Kaufmann, 2007. – 442 p.
3. GEROIMENKO, Vladimir − CHEN, Chaomei: Visualizing Information Using SVG and X3D. – London: Springer, 2005. – 298 p.
4. GKOUTZIS, Konstantinos: A Semantic Web based search engine with X3D visualisation of queries and results. – Plymouth: University of Plymouth, 2012. – 180 p.
5. GUARINI, Gianluca Daniele: HTML5 and CSS3 Transition, Transformation and Animation. – Birmingham: Packt Publishing, 2013. – 122 p.
6. KO, Chi Chung − CHENG, Chang Dong: Interactive Web-Based Virtual Reality with Java 3D. – Hershey: IGI Global, 2008. – 492 p.
7. KUNZ, Arthur: Web-3D-Welten systematisch erzeugen. – Hamburg: Diplomica Verlag, 2010. – 60 p.
8. POTENZIANI, Marco − CALLIERI, Marco − DELLEPIANE, Matteo: 3DHOP. In: Computers and Graphics, 2015. 52. sz., p. 129-141.
9. SUN, Fei − ZHANG, Zhaochuang: A lightweight and cross-platform Web3D system for casting process based on virtual reality technology using WebGL. In: International Journal of Advanced Manufacturing Technology, 2015. 80. sz., p. 801-816.
10. Web3D Consortium hivatalos honlapja http://www.web3d.org/

Beérkezett: 2017. I. 11-én.

Sirhán Bálint
a Debreceni Egyetem Informatikai Tudományok Doktori Iskola doktorandusza
E-mail: netkulcs@outlook.com