Eötvös Loránd Tudományegyetem
Informatikai Kar
Térképtudományi és Geoinformatikai Tanszék

AZ SVG FORMÁTUM, MINT ÁTHIDALÓ ESZKÖZ A TÉRINFORMATIKA, A GRAFIKA ÉS A WEBES FELÜLETEK KÖZÖTT

Tokai Tibor
térképész szakos hallgató

Témavezető:
Gede Mátyás
adjunktus

Budapest, 2013

Tartalomjegyzék

1. Bevezetés

Dolgozatom témájaként a kartográfiában és a számítógépes grafikában méltán népszerű vektoros grafikai formátum, az SVG bemutatását tűztem ki célul. Ennek a formátumnak óriási előnye, hogy a térképkészítésre, illetve megjelenítésre használt összes felület képes kezelni, ezáltal közös nevezőként funkcionálhat közöttük.

Számos alkalommal előfordulhatott már, hogy térképkészítés közben szerettünk volna egy bizonyos feladatot megoldani, viszont az általunk használt szoftverben erre nem volt lehetőség. Ilyen esetben szükség lehet arra, hogy a munkánkat egy olyan programban nyissuk meg, amiben ez a feladat könnyűszerrel megoldható. Erre az SVG használatával van lehetőség, hiszen ez a formátum kezelhető grafikai és térinformatikai programok által, valamint webes megjelenítésre is kiválóan alkalmas.

Fontos megemlíteni, hogy bár a szoftverek megnyitják az ilyen állományokat, a grafika a legtöbb esetben nem az elvárásaink szerint jelenik meg. A leggyakrabban fellépő hibák a szöveges elemeknél mutatkoznak, de előfordulnak koordináta-, illetve vonalvastagságbeli eltérések is. Ezek a problémák azért jelentkezhetnek, mert a különböző programok eltérő módon értelmezik az SVG fájlok nyelvezetét. Korrigálásuk rendszerint megoldható, de olyan is adódhat, hogy egy szoftvernél annyi hiba jelentkezik, hogy érdemesebb egy másikkal próbálkozni. Munkám során arra kerestem a választ, hogy a térképészetben használt szoftverek által készített SVG állományoknak milyen a szerkezete, milyen eltérések találhatóak bennük, és egy másik programban megnyitva milyen hibáik lehetnek.

Egy további említésre méltó téma ebben a kérdéskörben a térképek publikálása a weben. Az internet az emberek mindennapjainak részévé vált és a felhasználók száma az okostelefonok, illetve a táblagépek megjelenésével egyre csak növekszik. Az SVG, mint vektoros formátum ezen a területen is számos előnnyel rendelkezik, és napjainkban egyre népszerűbb a használata. Ebből kifolyólag lényegesnek tartottam megvizsgálni, hogy milyen támogatottsággal rendelkezik az állomány a különböző böngészőkben és a gyakorlatban ez hogyan mutatkozik. Az ilyen típusú fájlok praktikus internetes megjelenítéséhez létrehoztam egy szemléltető honlapot, ahol lehetőség van helyi állományokat megnyitni, vizsgálni azok rétegszerkezetét, illetve nagyítani a grafikát.

2. Vektoros és raszteres rendszerek a kartográfiában

A térképészetet területi egységekre bontva, két fő részről beszélhetünk. Az egyik a hagyományos, papír alapú kartográfia, a másik a geoinformatika vagy térinformatika. Vektoros és raszteres rendszerek mindkét területnél jelen vannak. Ha térinformatikáról beszélünk, akkor adatmodellekként említhetjük a két módszert, ugyanis lehetőség nyílik a térbeli vonatkozású adatokhoz különböző típusú attribútumparamétereket kapcsolni, vagyis egy modellt alkotni. Ebben az esetben kevésbé lényeges, hogy az adatokhoz kapcsolt térképek esztétikailag helyesek legyenek, inkább a térbeli elhelyezéshez nyújtanak segítséget. A hagyományos kartográfiában a két módszer már grafikailag értelmezendő, sokkal fontosabb maga a térkép, vagyis a térbeli adatok, objektumok térképi megjelenítésének a kérdésköre. (ELEK 2006)

Bizonyos felületeken, például egy weblap formájában, lehetőség van arra is, hogy egy esztétikailag igényesen elkészített, kartográfiailag helyes térképhez, a térinformatikához hasonlóan adatokat kapcsoljunk, és mindezt együtt jelenítsük meg. Ehhez igen komplex tudásra van szükség, hiszen amellett, hogy tisztában kell lenni azzal, hogyan lehet kartográfiailag helyes térképet szerkeszteni – amennyiben saját térkép felhasználására kerül sor –, ismerni a kell a térinformatikai adatmodellek működését, illetve nem utolsó sorban a honlapszerkesztés és a webprogramozás részleteit is.

2.1. Vektorgrafika és rasztergrafika

Egy térkép elkészültekor felmerülhet a kérdés, hogy vektoros vagy raszteres állományban érdemes-e menteni a munkát. A végső döntés annak fényében születhet meg, hogy milyen célra szeretnénk használni a grafikát. Ha nyomtatásra szánjuk, akkor sokszor elegendő valamilyen raszteres állománnyal dolgozunk, mint például a JPEG vagy a TIFF, de előfordulhat, hogy egy vektoros formátum használata is megfelelő, mint a PDF. Abban az esetben is legtöbbször elegendő valamilyen raszteres állomány alkalmazása, ha webes megjelenítésre szánjuk a munkát, viszont az utóbbi időben előtérbe került a vektoros állományok használata is ezen a területen, mert egyre fontosabbá válik a felbontás független honlapok készítése. Az előbbi kérdés a

kartográfiában is többször felmerülhet, hiszen a felhasználók számára nem mindegy, hogy egy térképet egy adott felbontásban publikálunk vagy pedig szabadon változtatható a felbontása. Ahhoz, hogy tisztában legyünk a két eljárás sajátosságaival, a következőkben rövid leírásban ismertetem azokat.

2.1.1. Rasztergrafika

Egy raszteres kép képpontokból vagy más néven pixelekből áll, amik mátrix-szerűen, szabályos szerkezetben építik fel a grafikát. Minden egyes képponthoz tartozik egy színérték vagy szürkeségi érték. Az különböző színű pixelek sokasága építi fel a grafikát. A kép minőségét annak felbontása határozza meg. A felbontás mértékegysége a ppi (pixels per inch) vagy a dpi (dots per inch), előbbit akkor szokás használni, ha munkánkat képernyőn szeretnénk megjeleníteni, utóbbit pedig akkor, ha nyomtatásra szánjuk. Minél nagyobb a felbontás értéke, annál szebb lesz a kép minősége, hiszen ilyenkor egy hüvelyknyi területre több képpont sűrűsödik össze.

A rasztergrafika előnye, hogy maga a kép gyorsan elkészíthető vagy feldolgozható, könnyen manipulálható, illetve egyszerű az adatszerkezete. Térképek esetében a gyors elkészíthetőség csak átvitt értelemben igaz, hiszen a szoftver valóban gyorsan generál egy raszteres képet, viszont az előtte lezajló rajzolási folyamat rengeteg munkaórát vehet igénybe. Ezekkel ellentétben, hátrányaként említhető, hogy számos esetben az adatállomány nagyméretű, ami webes alkalmazások esetében a megjelenítési időben gondokat okozhat, illetve tárolási szempontból is előnytelen lehet. Az ilyen módon készített képek rögzített felbontásúak, ebből adódóan a nagyításuk is meglehetősen korlátozott, ami szintén hátrányként értékelhető. [3]

2.1.2. Vektorgrafika

A vektorgrafika a rasztergrafikával ellentétben nem kis elemekből (képpontokból) építi fel a képet, hanem különböző geometriai alakzatokból. Minden egyes alakzat megjelenítése valamilyen matematikai transzformáció leképezésének eredményeképpen jön létre, helyzetüket a koordinátájuk határozza meg. Ilyen alakzatok például a pontok,

az egyenesek, a görbék, illetve a sokszögek. Az objektumokhoz különböző grafikai tulajdonságokat rendelhetünk, mint a szín, a vonalstílus, vagy a különböző textúrák. Fontos megemlíteni, hogy függetlenül attól, hogy a végső képet a képernyőre vagy nyomtatásra szánjuk, a vektoros grafika szinte mindig raszteressé konvertálódik át a megjelenítésnél. Ez alól kivétel, ha valamilyen vektoros állományként mentjük a grafikát, mert ilyenkor valóban vektoros képként és az arra jellemző tulajdonságokkal jelenítik meg az egyes felületek.

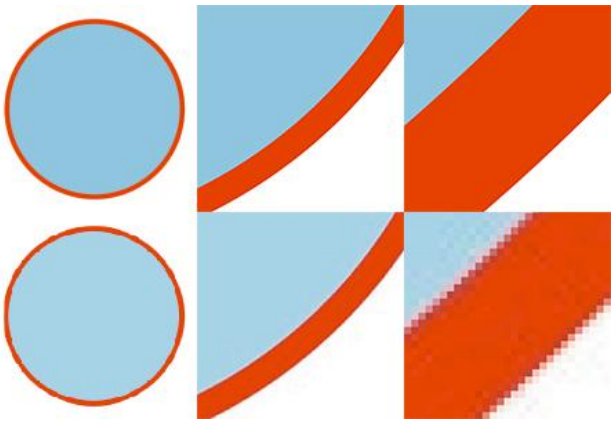
A vektorgrafikának számos előnye van a rasztergrafikához képest. Felbontás független, azaz tetszőlegesen lehet nagyítani vagy kicsinyíteni a képet torzulásmentesen. Lényegesen kisebb a memóriaigényük a raszterképekhez képest, illetve az elemek számos tulajdonsága, mint a méretük, irányuk, kitöltésük stb. utólagosan változtatható. Hátrányukként említhető, hogy az elkészítésük rendszerint sok időt vesz igénybe. Ez alól kivételt képezhet, ha például egy adatbázisból generálunk szintvonalrajzot, amit aztán vektoros állományként mentünk el, mert ez a folyamat rövid idő alatt is végbemehet. Egy másik előnytelen tulajdonsága pont a mértéktelen nagyítási lehetőségéből adódik, mert bizonyos grafikai hibák ilyenkor könnyebben észrevehetőek.

További előnye a vektoros állományoknak, azon belül is az SVG formátumnak, hogy JavaScript segítségével könnyen manipulálható. Ez magától értetődően a webes megjelenítéseknél számít rendkívül jó tulajdonságnak. Megfelelő hozzáértéssel a grafikánkat módosíthatjuk, interaktívvá tehetjük, de adott esetben akár egy teljes alkalmazási felületet építhetünk köré. Ahhoz, hogy ez lehetséges legyen, a programozási nyelv ismerete mellett, tisztában kell lenni az SVG nyelvezetével, amit a következő fejezetben mutatok be. [2]

2.1.3. Szemléltető példa a vektoros és raszteres megjelenítésre

A fentiekből kiderül, hogy talán a legnagyobb különbség a raszteres és a vektoros állományok között a felbontásukban rejlik. Ez leginkább a grafika nagyításánál mutatkozik. Az raszteres képek felbontása rögzített, a vektorosoké nem, és sokszor ennek a tulajdonságnak a hatására érdemes áttérni az utóbbi használatára. A szemléltetéséhez ugyanarról a képről készítettem egy raszteres (JPEG) és egy vektoros

(SVG) állományt. Az alábbi képen látható, hogy a két grafika a különböző nagyítási szinteken hogyan viselkedik. Rögtön szembetűnik, hogy a raszteres állomány egy bizonyos szint után pixelesedni, torzulni kezd, míg a vektoros állomány minden szinten ugyanolyan minőségben jelenik meg.



1. ábra – Vektoros és raszteres állományok különböző nagyítási szinteken

3. Az SVG állomány

Az SVG (Scalable Vector Graphics) egy XML alapú leíró nyelv, vagyis az általa kirajzolt grafikákat XML formátumban határozza meg. Egyre szélesebb körben használt kétdimenziós, statikus és mozgó vektorgrafikák meghatározására. Ennek megfelelően az elsajátításához szükség lehet némi előismeretre a HTML és az XML nyelvben. Három típusú grafikus elemet támogat: vektorgrafikus alakzatokat, rasztergrafikus képeket, illetve szövegeket. [1]

Az SVG fontos szerepet tölt be a világhálón a vektor alapú grafikák létrehozásában. Számos előnye közül, talán a legkiemelkedőbb a vektoros mivoltából ered, azaz nagyítás hatására sem torzul a grafika. Lehetőség van az SVG állománynak minden egyes elemét és attribútumát animálni, amely tovább színesítheti a megjelenítési élményt.

A nyelv jelen pillanatban W3C ajánlási státuszban van (W3C recommendation - REC), ami a World Wide Web Consortium (W3C) legelőkelőbb fejlesztési állapota. Ez azt jelenti, hogy a nyelv egy teljes elméleti és gyakorlati tesztelésen, illetve vizsgálaton ment keresztül a konzorcium által. Ennek eredményeként alapvető szabványként lett jóváhagyva, ami a támogatottság folyamatos növekedését eredményezi és lehetőséget nyújt más szabványokkal (DOM, XSL) való együttműködésre.

3.1. SVG állományok létrehozásának lehetőségei

SVG állományokat lényegében kétféleképpen hozhatunk létre. Az egyik mód, hogy valamilyen egyszerű szövegszerkesztő programot használunk a fájl létrehozására. Ilyenkor fontos pontosan ismerni az egész nyelvet, hiszen az összes rajzi elem leíró kódját magunktól kell begépelnünk. Ezt a módszert akkor érdemes alkalmazni, ha valamilyen egyszerűbb alakzatokat tartalmazó grafikát szeretnénk készíteni.

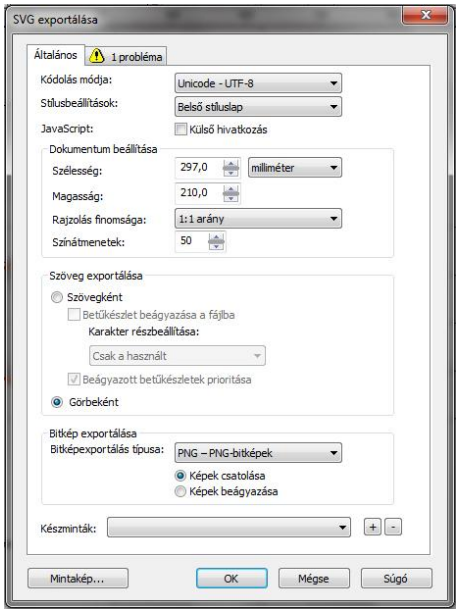
Egy komplexebb vektorgrafikus kép megalkotása azonban már rendkívül bonyolult, illetve időigényes feladat lenne az előző eljárással, mert ilyenkor a grafika pontossága mellett, a nyelv helyességére is ügyelni kell. Ebben az esetben az állomány létrehozására érdemesebb egy többek között erre a célra is fejlesztett grafikai program, vagy adatbázisok közreműködésével egy térinfomatikai szoftver használata.

A grafikai szoftverek eszközei segítségével könnyen megrajzolható az összes általunk szükségesnek vélt alakzat, amelyek összessége alkotja majd a kész térképi grafikát. Az elkészült rajzunkat a program exportálási funkciójával rengeteg féle raszteres, illetve vektoros formátumba, többek között SVG-be is konvertálhatjuk.

Ilyen programból ma már több is található a piacon. Létezik ingyenesen letölthető és használható, úgynevezett szabad szoftver, de igen drága, fizetős változatok is megtalálhatóak. SVG fájlok készítéséhez sokszor elég az ingyenes szoftver használata, de bizonyos esetekben szükségünk lehet a fizetős programok által nyújtott felhasználói élményre. Az következőkben ezek a szoftverek közül mutatok be néhányat, amelyek a kartográfiában is közkedveltek.

3.1.1. CorelDraw

A CorelDraw egy népszerű vektorgrafikus szerkesztő program, amelyet a kanadai Corel Corporation fejleszt Ottawaban. A Corel's Graphics Suite szoftvercsomag tagja, ami több másik grafikai programot is magába foglal. Jelenlegi legfrissebb X6-os verzióját, 2012 márciusában adták ki. Talán az egyik legnépszerűbb térképrajzolásra is használt fizetős szoftvernek tekinthető a honi kartográfiában. Számos előnye mellett rendkívül könnyen lehet vele többféle SVG állományt is létrehozni, amelyek későbbi kezelése az egyik legoptimálisabbnak mondható. [6]

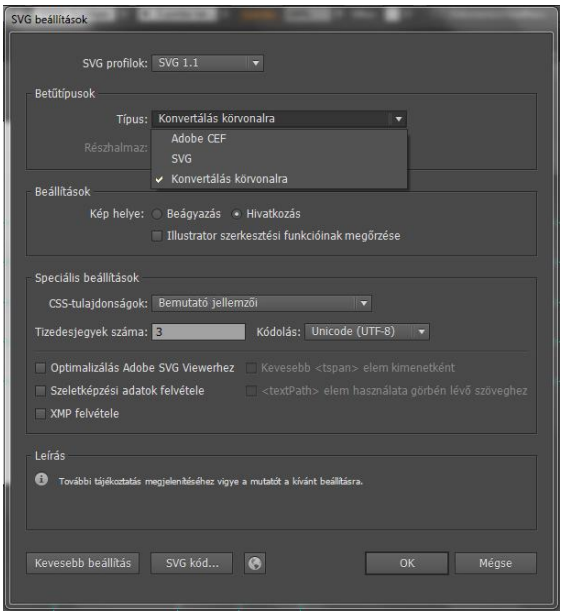


2. ábra – SVG exportálása CorelDraw-ban

Ahogyan az exportálási ablakban is látszik, igen sok féle beállítási lehetőség közül lehet választani az állomány elkészítéséhez. A karakterkódolás, a stílusok, illetve a dokumentum méretének beállítása mellett, itt lehet meghatározni azt is, hogy a képünkön szereplő szövegeket milyen módon exportálja a program. Választhatjuk, hogy egyszerű szövegként, a betűkészlet beágyazásával, vagy pedig görbékké konvertálva értelmezze és mentse a szoftver a szöveges elemeket. Ennek a fontossága a későbbi felhasználás során derül ki, mert az egyes felületek sok esetben különböző módon várják el az SVG-től a szövegkezelést, ezáltal a megjelenítés sem működik mindig olajozottan.

3.1.2. Adobe Illustrator

Az Adobe Illustrator a jelenlegi legnépszerűbb fizetős vektorgrafikus szerkesztő program a világon. Az Adobe Systems által fejlesztett szoftver, aminek a neve alá tartoznak többek között olyan nagynevű programok, mint az Adobe Photoshop, Flash, vagy Dreamweaver. A CS6-os (Creative Suite) termékcsalád tagjaként szerepel a legfrissebb változata, amely az Adobe által fejlesztett szoftvercsomag 16. kiadásaként említhető. A program rendkívül hasznos mindenféle grafikai munka elkészítésére, a webdizájner munkáktól, a logótervezésen át, egészen a térképekig. Alapvető fájlformátuma az *.ai*, de emellett rengeteg más vektoros, illetve raszteres formátumban képes elmenteni grafikánkat, mint például a *.psd, .pdf, .jpg, .png*, és természetesen az *.svg* is. SVG fájl készítése során az exportálási opciók szinte megegyeznek a CorelDraw-nál említettekkel. [5]



3. ábra – SVG exportálása Adobe Illustratorban

Szembetűnő különbségnek mondható, hogy ennél a programnál lehetőség van választani különböző SVG profilok között, amelyek az exportált fájl XML dokumentumtípus-definícióját adják meg. Vannak elsősorban asztali számítógéphez (SVG 1.0 és SVG 1.1), közepes teljesítményű eszközökre, például táblagépekre (SVG Basic 1.1), kisméretű eszközökre, mint mobiltelefonokra (SVG Tiny 1.1 és SVG Tiny 1.1+), illetve egyszerre több eszközre (SVG Tiny 1.2) tervezett verziók. Az egyes profilok hátulütője, hogy számtalanszor nem támogatnak alapvető SVG tulajdonságokat (különböző hatásokat, szimbólumokat, mintázatokat, szöveg tulajdonságokat) és annak ellenére, hogy kifejezetten bizonyos eszközökhöz vannak tervezve, a gyártó nem garantálja azok támogatottságát. Éppen ezért, talán legbiztonságosabb az alapvető SVG verzió használata, habár asztali gépekre szánták, mégis szinte minden eszközön megfelelően megjelenik.

3.1.3. Inkscape

Az InkScape egy ingyenes, nyílt forráskódú vektorgrafikus szerkesztő program, amelynek célja, hogy a különböző platformokon felvegye a versenyt a legnagyobb grafikai szoftverekkel. Fejlesztése 2003 óta tart, a legfrissebb verziója a 0.48.4-es. Különösen weboldalakhoz készült munkáknál közkedvelt, jól használható különböző

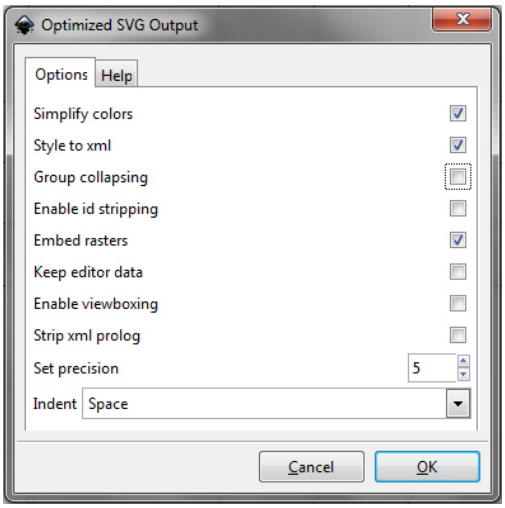
logók, ikonok, piktogramok tervezésére, de akár térképek szerkesztésére is kitűnően megfelel.

Alapvető kiterjesztése az *.svg* (inkscape SVG), ami teljesen megfelel az SVG által előírt szabályoknak, csupán abban különbözik egy átlagos ilyen típusú fájltól, hogy az Inkscape névterét használja néhány extra információ tárolására. Emellett több másik SVG formátumban is menthetjük munkánkat (plain, compressed, compressed plain, optimized). A különbség az exportálható változatok között nem a grafikában mutatkozik, hanem a fájl nyelvezete és mérete lesz más. [7]

A ’plain’, vagyis egyszerű SVG, egy olyan letisztult formátum, ami abszolút mentes az egyes programok sajátos sallangjaitól, ezáltal elméletben kompatibilis minden grafikai szoftverrel. Az ’optimized’, ahogyan a neve is mutatja, egy optimalizált változat a felhasználás során használt felületekhez, különösen az internetes böngészőkhöz. A ’compressed’ változatok erősen tömörítettek, minden elhanyagolható elem kimarad belőlük, emberi szemmel tulajdonképpen olvashatatlanok.

Forráskódjaik mellett, a fájlok méretében is észrevehetők különbségek. A legkisebb méretűek a tömörítettségük miatt a ’compressed’ állományok, a legnagyobbak pedig az alap ’inkscape SVG’ fájlok, amelyek rengeteg, nem feltétlenül szükséges információt tartalmaznak. A típusok közötti különbség akkor lehet fontos, ha a rajzunkat egy másik szoftverben vagy böngészőben szeretnénk megnyitni, illetve ha valamilyen programozási nyelvvel szeretnénk manipulálni.

Az ’optimized’ SVG fájl exportálása során többféle beállítási lehetőség közül választhatunk. Átkonvertálhatjuk az összes színt #RRGGBB, vagyis hexadecimális formátumba (Simplify colors), a stílusokat megadhatjuk XML attribútumként (Style to xml), eltávolíthatjuk az összes hivatkozás nélküli `id` attribútumot (Enable id stripping), a raszteres adatokat tárolhatjuk base64 tartalomkódolási formában (Embed rasters), illetve megtarthatjuk a grafikus szerkesztő program (Inkscape, Adobe Illustrator) sajátos elemeit és attribútumait (Keep editor data).



4. ábra – Optimized SVG fájl beállítási lehetőségei exportálásnál Inkscape-ben

3.2. Az SVG állomány általános felépítése

Az SVG dokumentumok sok esetben bizonyos szempontokból különbözhetnek egymástól, attól függően, hogy éppen milyen szoftverrel hoztuk létre az adott állományt, viszont alapszerkezetük mindig megegyezik. Általánosságban elmondható, hogy mindegyik dokumentum tartalmazza az következő egységeket:

- Technikai és dokumentációs adatok (deklaráció)
- Előre definiált alakzatok
- Stílusdefiníció

A deklarációs rész az állomány fejlécében található és rendszerint így néz ki:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
```

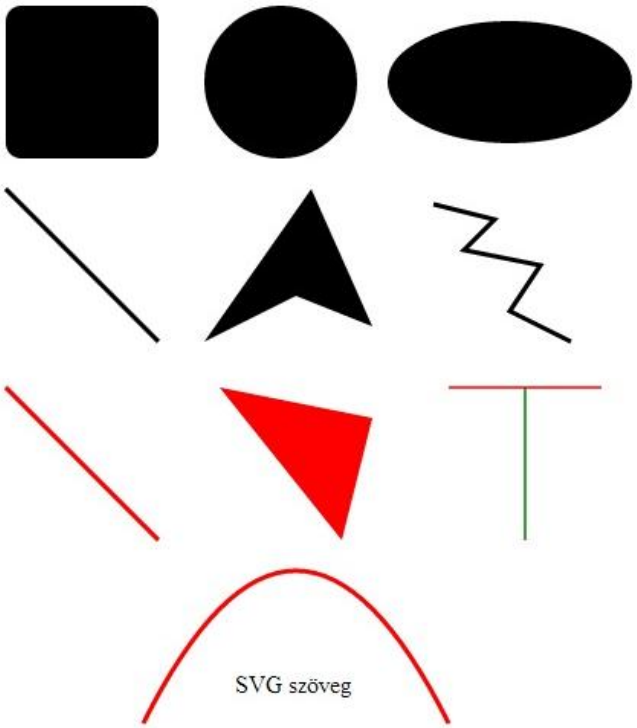
A kód legelején tájékoztatjuk a böngészőt, hogy 1.0 verziójú XML dokumentumot fogunk használni, majd megadjuk a karakterkódolást. Ez után a dokumentumtípus-definícióval (DTD) utalunk rá, hogy a böngésző hol találhat érvényességi előírásokat a dokumentumunkhoz. Az SVG dokumentum törzse egy nyitó `<svg>` és egy záró `</svg>` elem közé kerül. Itt hozhatjuk létre a különböző alakzatokat, szövegeket, azaz a grafikát

alkotó elemeket. A nyitóelem `xmlns` attribútuma határozza meg a névteret, a `version` pedig a verziószámot.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<!-- Ide kerül a grafikát alkotó elemek leírása -->
</svg>
```

3.2.1. Alakzatok

Az SVG-ben, mivel vektoros formátumról lévén szó, minden rajz geometriai alakzatokból áll össze. Ezek az alakzatok előre definiáltak a nyelvben és a hozzájuk tartozó elemek segítségével könnyedén használhatóak a fejlesztések során. Amellett, hogy kirajzoljuk magukat a formákat, stílusok segítségével a megjelenésüket is alakíthatjuk. Minden egyes elemnek vannak attribútumai, amelyekkel a méretük vagy az elhelyezkedésük változtatható. Az alábbi komplex példában bemutatom a legalapvetőbb alakzatokat néhány szükséges stílus meghatározással az átláthatóság kedvéért:



5. ábra – Alapvető alakzatok az SVG nyelvben

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="450px"
height="500px">
  <g id="vegyes_alakzatok">
    <rect x="20" y="20" rx="10" ry="10" width="100" height="100" />
    <circle cx="200" cy="70" r="50" />
    <ellipse cx="350" cy="70" rx="80" ry="40" />
    <line x1="120" y1="240" x2="20" y2="140"
style="stroke:rgb(0,0,0);stroke-width:3" />
    <polygon points="220,140 260,230 210,210 150,240" />
    <polyline points="300,150 340,160 320,180 370,190 350,220 390,240"
style="fill:none;stroke:black;stroke-width:3" />
  </g>
  <g id="path_alakzatok">
    <path d="M 20 270 L 120 370 Z" stroke="red" stroke-width="3" />
    <path d="M 160 270 L 240 370 L 260 290 Z" fill="red"/>
    <path d="M 310 270 H 410 Z" stroke="red" />
    <path d="M 360 270 V 370 Z" stroke="green" />
    <path d="M 110 490 q 100 -200 200 0" stroke="red" stroke-width="3"
fill="none" />
  </g>
  <g id="szoveg">
    <text x="170" y="470">SVG szöveg</text>
  </g>
</svg>
```

Az előző példában az összes alapvető alakzat megtalálható, amiket listaszerűen is bemutatok, feltüntetve a hozzájuk tartozó elemeket is:

- Négyszög, `<rect>`
- Kör, `<circle>`
- Ellipszis, `<ellipse>`
- Vonal, `<line>`
- Sokszög, `<polygon>`
- Törött vonal, `<polyline>`
- Görbe, `<path>`
- Szöveg, `<text>`

Ezeket az alakzatokat három csoportra bontottam a mintafájl elkészítésekor. A felosztáshoz a nyelv `<g>` elemét használtam fel. Ez az elem a grafikai programokból jól ismert rétegekhez hasonlóan működik és előnye, hogy a benne lévő alakzatokat egy egységként kezeli. Ez fontos lehet a különböző stílusdefiníciók meghatározásakor, továbbá akkor is, ha egy összetettebb objektumot több alkalommal szeretnénk

felhasználni. Az első csoportba vegyes elemek találhatóak, amelyeket fekete színnel jelöltem. A második rétegre a görbe vagy `<path>` elem segítségével elkészített alakzatokat csoportosítottam, ezek vörös, illetve egy esetben zöld színűek. Az utolsó csoportba a példaszöveg található.

Mint látható, az alakzatok elhelyezéséhez minden esetben tartozik egy vagy több koordinátapár, attól függően, hogy az objektum mely részének a helyzetét kell meghatároznunk. Négyszög, kör vagy ellipszis esetében elég egy koordinátapárt megadnunk. Négyszögnél a bal felső sarok koordinátáját (`x, y`), körnél és ellipszisnél a középpontok koordinátáját (`cx, cy`) kell megadnunk. Vonalnál a kezdő és a végpont (`x1, y1` és `x2, y2`), sokszögnél és törött vonalnál minden egyes pont koordinátáját szükséges meghatároznunk. Négyszögnél szükséges még a szélesség és a magasság megadása (`width, height`), illetve opcionálisan beállítható a sarkok lekerekítése is (`rx, ry`). A kör és az ellipszis méretének a meghatározásához, előbbinél az átmérő (`r`), utóbbinál a fél nagy- és kistengely számszerű definiálása a fontos (`rx, ry`).

A második rétegen található görbe vagy `<path>` elemek sokkal komplexebbek, mint az alap alakzatok és használatukkal gyakorlatilag az összes első csoportos elem létrehozható. Ennek megfelelően létrehozhatunk velük egyenes és törött vonalak, poligonokat, ellipszisíveket, valamint másod- és harmadfokú Bézier-görbéket is. Az alakzatok leírására a `<path>` elem `d` attribútuma szolgál. Keretein belül többféle utasítást is adhatunk a kívánt forma megrajzolásához. Ezek az utasítások a következők lehetnek:

- **`M`** (moveto) – Helyzet megadása rajzolás nélkül (kiindulási pontnak)
- **`L`** (lineto) – Vonal rajzolása adott pontig
- **`H`** (horizontal lineto) – Vízszintes vonal rajzolása adott pontig
- **`V`** (vertical lineto) – Függőleges vonal rajzolása adott pontig
- **`C`** (curveto) – Harmadfokú Bézier-görbe rajzolása
- **`S`** (smooth curveto) – Simított harmadfokú Bézier-görbe rajzolása
- **`Q`** (quadratic Bézier curve) - Másodfokú Bézier-görbe rajzolása
- **`T`** (smooth quadratic Bézier curveto) – Simított másodfokú Bézier-görbe rajzolása
- **`A`** (elliptical Arc) – Ellipszisív rajzolása
- **`Z`** (closepath) – Görbe lezárása

Minden utasítás mellé szükséges egy vagy több koordinátapárt is megadni, hiszen így határozhatjuk meg, hogy a rajzolni kívánt görbe vagy egyenes honnan induljon, illetve meddig tartson. Egyenesek esetében elegendő megadni a végpontot, Bézier-görbéknél azonban szükséges a kezdő- és végpont, valamint a köztes ellenőrzési pont(ok) definiálása is.

Az utolsó csoportban a szöveg vagy `<text>` elem bemutatására egy egyszerű eset látható. Az elem attribútumaként az elhelyezéséhez szükséges koordináták (`x, y`) vannak megadva, a nyitó- és záróelem között pedig az a szöveg, ami az ábrán is megjelenik.

Szükség szerint lehetőség van ferdén írott vagy görbített szövegek létrehozására is, ami térképek esetében sűrűn előfordulhat. Előbbinél a `<text>` elem `transform` attribútumában kell megadni a `rotate` parancsot, azon belül is egy számértékkel az elforgatás szögét. Az utóbbinál egy előre definiált görbe mentén van lehetőség elhelyezni a szöveget. Itt a `<text>` elem mellett szükség van egy `<textPath>` elnevezésű elemre is, amelynek segítségével hozzá lehet kapcsolni a szöveget a kívánt görbéhez.

```
<text x="15" y="15" fill="red" transform="rotate(45 10,30)">Példa
szöveg</text>
```


```
<defs>
   <path id="path1" d="M75,20 a1,1 0 0,0 115,0" />
</defs>
<text x="10" y="100" style="fill:red;">
   <textPath xlink:href="#path1">Példa szöveg Példa szöveg</textPath>
</text>
```

Fontos megemlíteni, hogy természetesen nem csak szövegeket, hanem bármilyen más alakzatot is lehetőség van elforgatni a `transform` attribútum használatával.

Ahogyan az összetett példában is látható volt, bizonyos objektumok egyéni kitöltési színnel vagy vonalvastagsággal jelentek meg. Az SVG-ben lehetőség nyílik az elemek létrehozása és elhelyezése mellett, stílusuknak a megváltoztatására is. Ezt többféleképpen tehetjük meg, a következőkben ezeket a lehetőségeket ismertetem.

3.2.2. Stílusdefiníciók

Ha valaki már jártas a honlapkészítésben, akkor biztosan ismerős neki a CSS (Cascading Style Sheets) stílusleíró nyelv, amellyel HTML oldalak megjelenését írhatjuk le. Használatával lehetőség nyílik minden egyes elem színét, méretét vagy átlátszóságát változtatni, de akár meghatározott eseményekhez animációkat is készíthetünk a segítségével. Az SVG esetében is hasonló módon járhatunk el, stílusdefiníciók megadásával minden egyes alakzatnak egyszerűen változtathatjuk a megjelenését, osztályok bevezetésével pedig egy stílusleírást akár több elemhez is rendelhetünk.

Stílusok hozzárendelése a dokumentumhoz vagy egyes elemekhez többféle módon történhet:

- Közvetlen stílusmegadás attribútumként
- Közvetlen stílusmegadás a `style` attribútum segítségével
- Belső stíluslap használatával
- Külső stíluslap használatával

Az első két eset a legegyszerűbb, ezeket akkor érdemes használni, ha elemeink száma kevés, vagy ha stílusaik nem túl változatosak. Ha egy stílust attribútumként adunk meg, akkor az adott tulajdonság nevét és értékét kell használnunk az attribútum megfelelő helyeire. A második esetben a `style` attribútumot használjuk a stílusok meghatározására és a különböző tulajdonságokat ennek értékeként adjuk meg:

```
<line x1="10" y1="10" x2="50" y2="50" stroke="red" stroke-width="3" />

<line x1="10" y1="10" x2="50" y2="50" style="stroke: red; stroke-
width: 3" />
```

A másik két módszernél úgynevezett stíluslapokat használunk, amelyek magukba foglalják a különböző stílusleírásokat. A belső stíluslapot a dokumentumon belül kell

elhelyezni a HTML-ből jól ismert `<style>` elemben egy `<![CDATA[ … ]]>` szakaszban (character data). Erre a megkülönböztetett részre azért van szükség, mert a stílusleírás nem felel meg az XML által felállított szabályoknak és nélküle a böngésző nem tudná jól értelmezni az állományunknak ezt a részét. Ez a módszer akkor javasolt, ha több olyan alakzatunk is van egy dokumentumon belül, amit ugyanolyan stílussal szeretnénk ellátni. Az alakzatokra az adott elem nevével vagy egy előre definiált osztálynévvel (`class`), illetve azonosítóval (`id`) lehet hivatkozni.

```
<style type="text/css">
   <![CDATA[
      rect {
         fill: green;
         stroke: red;
      }
      .korok {
         stroke: #666;
         stroke-width: 3;
      }

   ]]>
</style>
```

Külső stíluslapoknál az alakzatokra való hivatkozás ugyanígy történik, a különbség csupán annyi, hogy maga a stílusleírás egy *.css* kiterjesztésű fájlban található, és ezáltal nincs szükség az előbb említett megkülönböztető szakaszra sem. Az eljárásnak a használata akkor célszerű, ha több SVG állománnyal dolgozunk és mindegyikhez egy közös stíluslapot szeretnénk használni. A külső stíluslapra való hivatkozásnál meg kell adnunk annak pontos elérési útját:

```
<?xml-stylesheet href="styles/kulso_stiluslap.css" type="text/css"?>
```

Stílusok meghatározásakor voltaképpen az elemek tulajdonságaihoz tartozó paraméterek megadását értjük. Beállíthatjuk, hogy az alakzatok egyes jellemzői milyen színűek, vastagságúak, vagy egyéb más megjelenésűek legyenek. A térképek és minden más grafika létrehozásakor az alábbi tulajdonságok változtatása a legjellemzőbb:

- **`stroke:`** vonal színe – szín nevével (black) vagy hexadecimálisan (#000000)
- **`stroke-width:`** vonal vastagsága – tetszőleges pozitív számmal
- **`stroke-opacity:`** vonal átlátszósága – 0 (teljesen átlátszó) és 1 (teljesen átlátszatlan) között

- **`stroke-linecap:`** vonalvégződés – butt (levágott), round (lekerekített) vagy square (négyzetes)
- **`stroke-dasharray:`** vonal szaggatottsága – vonalak és az őket elválasztó üres helyek hosszának megadásával
- **`fill:`** kitöltési szín – szín nevével (black) vagy hexadecimálisan (#000000)
- **`fill-opacity:`** alakzat (poligon) átlátszósága – 0 (teljesen átlátszó) és 1 (teljesen átlátszatlan) között

Ha egy grafikai programot használunk SVG fájl létrehozására, akkor a rajzunkhoz tartozó stílusok meghatározásának módjáról a szoftver gondoskodik. Különböző programoknál a stílusok definiálása más és más módon történhet, ezt mindig az adott fejlesztő cég dönti el, hogy melyik módszert alkalmazza. A CorelDraw által exportált fájl mindig belső stíluslapot használ, az Adobe Illustrator esetében a tulajdonságok attribútumként való használata a jellemző. Inkscape-nél más a helyzet, itt az exportálási beállításoktól függően a stílusmegadás történhet a `style` attribútum segítségével vagy a tulajdonságok attribútumként való megadásával is. Előbbinek a használata vagy az Inkscape által automatikusan generált SVG fájlban vagy a választható ’plain SVG’ fájlban fordul elő, utóbbi pedig a szintén opcionálisan választható ’optimized SVG’ fájl esetében látható. Természetesen attól, hogy a stílusok különbözőképpen vannak meghatározva, a grafika megjelenése nem változik, viszont eseteként az állomány forráskódja áttekinthetőbb lesz.

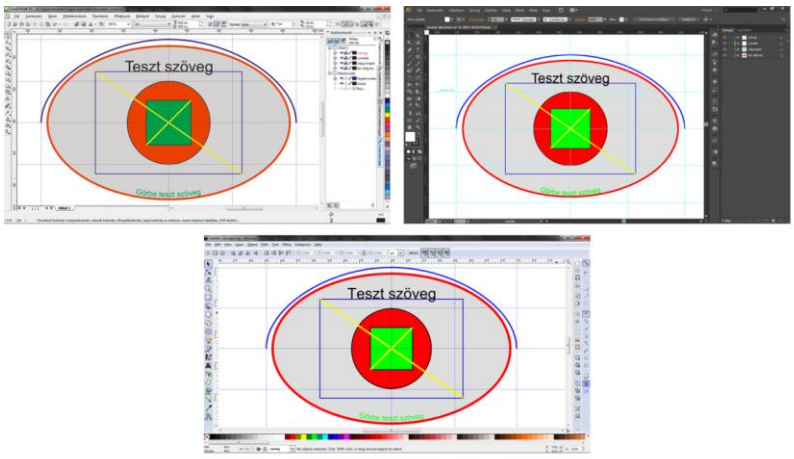
Már ebből a példából is látszik, hogy az egyes szoftverek milyen változatosan állíthatnak elő SVG fájlokat, pedig itt még csak a stílusmegadásokat említettem. Ez a későbbi felhasználás szempontjából általában nem előnyös, ezért fontos lehet a többi esetlegesen előforduló különbözőség ismerete is. [8] [11]

4. SVG állományok tesztelése

Ahogyan már korábban is említettem, SVG állományokat többféle szoftverrel is elő lehet állítani, legyen az grafikai vagy térinformatikai. Ebből kifolyólag a fájlok nyelvezete is igen különböző lehet, annak ellenére, hogy ugyanolyan nyelven íródtak. Ez esetenként gondot okozhat, ha egy adott szoftverben előállított fájlt egy másik programban szeretnénk megnyitni, vagy ha az SVG állományunkat webes célokra szeretnénk felhasználni.

4.1. SVG a különböző grafikai szoftverekben

Munkám során többféle SVG fájlt is elkészítettem az egyes grafikai programokkal, hogy kipróbáljam melyik változat működik a legjobban minden felületen. Ehhez mindegyik szoftverben megrajzoltam egy-egy a lehetőségekhez mérten teljesen megegyező grafikát, ugyanolyan színek, méretek, illetve vonalvastagságok megadásával. Ezeket az állományokat a teszteléshez hoztam létre, ezért törekedtem az egyszerűségre, hogy minél könnyebben lehessen velük szemléltetni a különbségeket. Azért nem kész térképi grafikákkal dolgoztam, mert egyrészt igen nehéz lett volna teljesen megegyező térképeket szerkeszteni a különböző szoftverekben, másrészt bonyolultságukból fakadóan az exportált fájlok forráskódjai is túlságosan összetettek lettek volna, ami nem optimális a későbbi szemléltetéshez. Az elkészült grafikák alább láthatóak:



6. ábra – Három különböző szoftverrel elkészített grafika (CorelDraw, Adobe Illustrator, Inkscape)

A három rajz a legalapvetőbb alakzatokat tartalmazza, mint a négyszög, a kör, az ellipszis, az egyenes vonal, illetve a görbe (Bézier-görbe). RGB színprofillal dolgoztam, mert kizárólag képernyőn való megjelenítésre szánom az állományokat. Első ránézésre teljesen egyformák a grafikák, azonban alaposabb vizsgálat után látszik, hogy a görbe mentén futó szövegekben apró elhelyezkedési és futásbeli különbségek vannak. Ennek oka, hogy mindegyik programban másmilyen technikával lehet létrehozni görbített szöveget, ami megnehezíti az összeegyeztetésüket. Ez persze döntően nem befolyásolja az ebből elkészített SVG állományok szerkezetét, csak koordinátabeli különbségeket eredményezhet.

Exportálás során legfőképpen a szöveges elemek kezelése között lehetnek eltérések. Szöveget exportálhatunk egyszerű vagy saját beágyazott betűkészlettel rendelkező szövegként, valamint görbeként. Ebből kifolyólag egy programmal több állományt is létre kellett hoznom, mert nem tudhattam előre, hogy melyik lesz a későbbi felhasználáshoz a legoptimálisabb változat. Három állományt készítettem el szoftverenként, tehát összesen kilenc exportált fájlom lett. CorelDraw-nál és Adobe Illustratornál a szövegbeállítások miatt volt szükség erre a mennyiségre, Inkscape-nél pedig az előre meghatározott SVG fájlok sokasága miatt.

4.1.1. Fájlszerkezet CorelDraw-nál

A CorelDraw-val elkészített állományaim, az előzetes beállításoknak köszönhetően, belső stíluslappal rendelkeznek, amiben különböző osztályokba vannak rendszerezve az egyes alakzatok stílusainak leírásához szükséges tulajdonságok. Az itt definiált osztályok az objektumok megrajzolásáért felelős elemeknél vannak meghívva. Az exportált fájlokban `<g>` elemek reprezentálják a rajzolás közben elkészített rétegeket, azok nevei pedig azonosítóként (`id`) lettek hozzárendelve ezekhez az elemekhez. A program a rétegek neveiben nem tudja megfelelően kezelni a szóközt exportálás során, ilyenkor az azonosítóban egy erre a célra definiált karakterlánccal helyettesíti azt (pl.: `<g id="kör_x0020_ellipszis">`). A kész állományok közötti különbséget végeredményben az előbb említett szövegbeállítások befolyásolják.

A legegyszerűbb fájlszerkezetet és ezzel együtt a legkisebb fájlméretet, az egyszerű szövegként való exportálás során kaptam. Ebben a betűkészlet a

stílusdefiníción belül egy osztályként van meghatározva, amit később a <text> elemhez hozzárendel a program. A görbe mentén futó szöveg esetében minden betűt külön <text> elemben tárol, amin belül egy <tspan> elem rotate attribútumával adja meg az elforgatás mértékét.

A beágyazott betűkészlettel felvértezett állománynál már nőtt a forráskód hossza és a fájl mérete, ami annak köszönhető, hogy az előző exportálási módhoz képest létrejövő fájlszerkezet még kibővült egy külön résszel, amelyben állományon belül lett definiálva a betűkészlet is. Itt minden egyes karakter, ami a szövegben szerepel, görbeként van meghatározva, de csak egyszer, függetlenül attól, hogy hányszor fordul elő a szövegben.

A leghosszabb forráskódot és a legnagyobb fájlméretet a szöveg görbeként való exportálása adta. Itt a normális, illetve a görbe mentén futó szöveg esetében is minden karaktert egy görbével ír le a program, a <path> elem segítségével. A legfőbb oka ez volt annak, hogy ennél a módszernél a leghosszabb a forráskód, ugyanis ebben az esetben akkor is újradefiniál egy karaktert a program, ha az már korábban előfordult a szövegben.

4.1.2. Fájlszerkezet Adobe Illustratornál

Adobe Illustratorral készített állományok esetében a stílusok attribútumként vannak meghatározva közvetlen stílusmegadással. A rétegek nevei itt is a <g> elemekben jelennek meg, a szóközöket pedig alulvonással helyettesíti a program.

A szövegeket menthetjük Adobe CEF formátumban, amely a rendszer betűtípus adatait használja a kisméretű betűk leképezésére, illetve SVG-ként és körvonalként (görbeként) is. A hivatalos Illustrator dokumentáció szerint az utolsó két esetet minden SVG-megtekintő támogatja, tehát ajánlatosabb a használatuk, az első viszont az Adobe SVG Viewer kivételével csak helyenként élvezhet támogatottságot. Ennek valóssága azért kérdőjelezhető meg, mert exportálás után azt tapasztaltam, hogy az Adobe CEF és az SVG típus választása esetén az elkészült fájlok forráskódja teljesen megegyezik, tehát valószínűleg megjelenítésnél is hasonlóan kell viselkedniük. Következtetésképpen a későbbiekben egy egységként fogom őket említeni.

Adobe CEF és SVG típusú exportálás esetén a szöveges elemek kétféleképpen kerülnek leírásra. A normál szöveget egyszerűen a `<text>` elem segítségével hozza létre a program, amiben attribútumként szerepel a betűcsalád és a betűméret is. A görbe mentén futó szövegnél a karakterek `<path>` elemekkel vannak meghatározva, tehát görbeként jönnek létre. Körvonalként való exportálás esetén a normál szövegek is görbeként jönnek létre.

4.1.3. Fájlszerkezet Inkscape-nél

Inkscape esetében is három fájlt exportáltam. Ezeknek az állományoknak a forráskódja nagymértékben eltér a másik két szoftverben létrehozott SVG fájlokétól. A program sokszor a saját névterét használja bizonyos dolgok leírására, de ennek elemzésére nem feltétlen van szükség. Néhány dolog azonban itt is megfigyelhető a különböző típusú SVG fájlokban.

Az alap ’inkscape SVG’ és a ’plain SVG’ esetében közvetlen stílusmegadást alkalmaz a program a `style` attribútum segítségével, míg az ’optimized SVG’-nél, az exportálási beállításoktól függetlenül, mindig közvetlenül a tulajdonság nevét használja attribútumként a stílus megadására. A másik két szoftverrel ellentétben, itt nem egyszerű azonosítóként szerepel a rétegek neve a `<g>` elemekben, hanem az `inkscape:label` sajátos névterű attribútumon belül (pl.: `inkscape:label="kör ellipszis"`), ami a későbbi felhasználásnál nem feltétlenül kedvező.

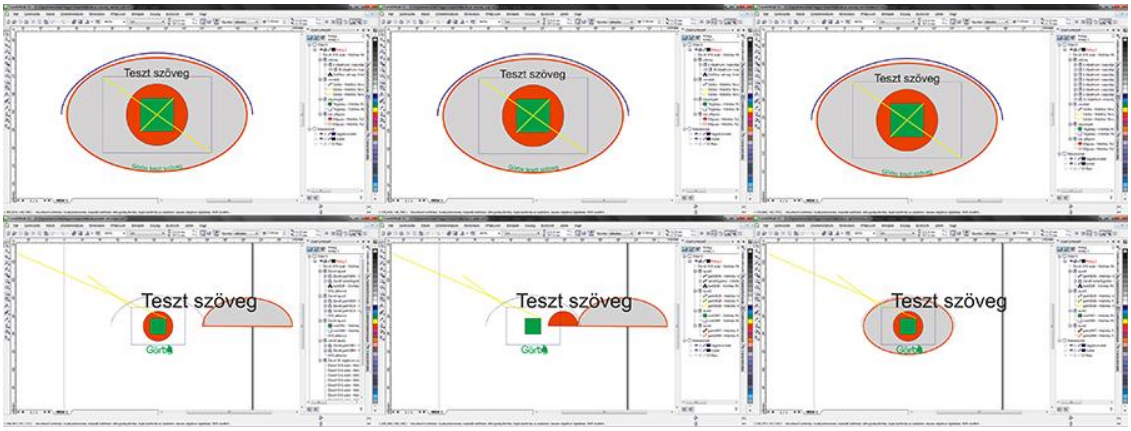
A szövegek kezelése kifejezetten előnyösnek mondható az előzőekhez képest, ugyanis itt a normál, illetve a görbe mentén futó szöveg esetében is egyszerűen a `<text>` elemet alkalmazza a program. Görbe szövegnél ezt a `<textPath>` elemmel kiegészítve oldja meg a szoftver, így a lehető legrövidebben, néhány sorral leírható a grafikának ez a része is. Legfőképpen ennek is köszönhetően mindegyik Inkscape által létrehozott SVG fájl mérete igen kicsinek mondható.

Az ’optimized SVG’ állománynál az exportálási beállítások között a ’Keep editor data’ opció bekapcsolását tartottam leginkább fontosnak. Abban az esetben, ha nincs bekapcsolva ez a lehetőség, akkor a program számos más információ mellett, nem szerepelteti a forráskódban azt az attribútumot sem, ami a rajzoláskor létrehozott

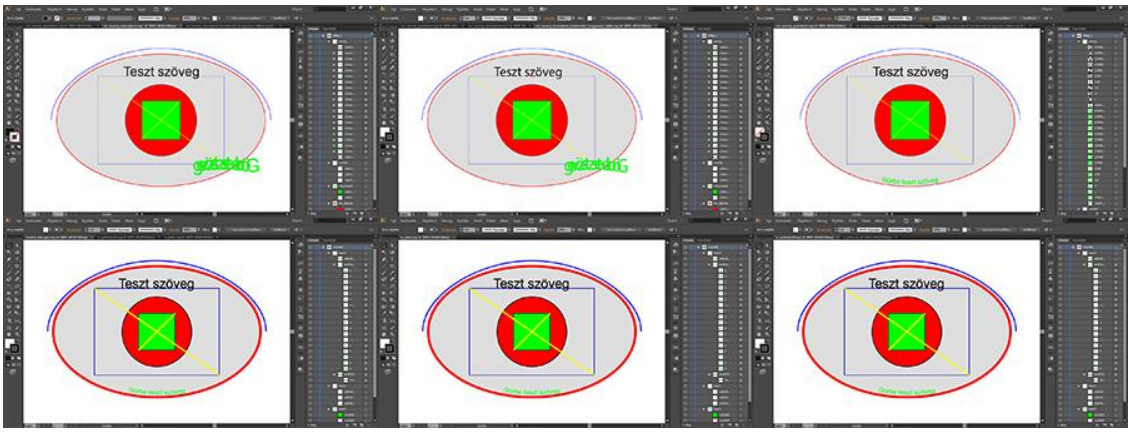
rétegek neveit tartalmazza. Így maga a forráskód összetettebb lesz, viszont szerepel benne ez az információ, ami később még fontos is lehet.

4.1.4. Állományok megnyitása az exportáló szoftvertől eltérő programban

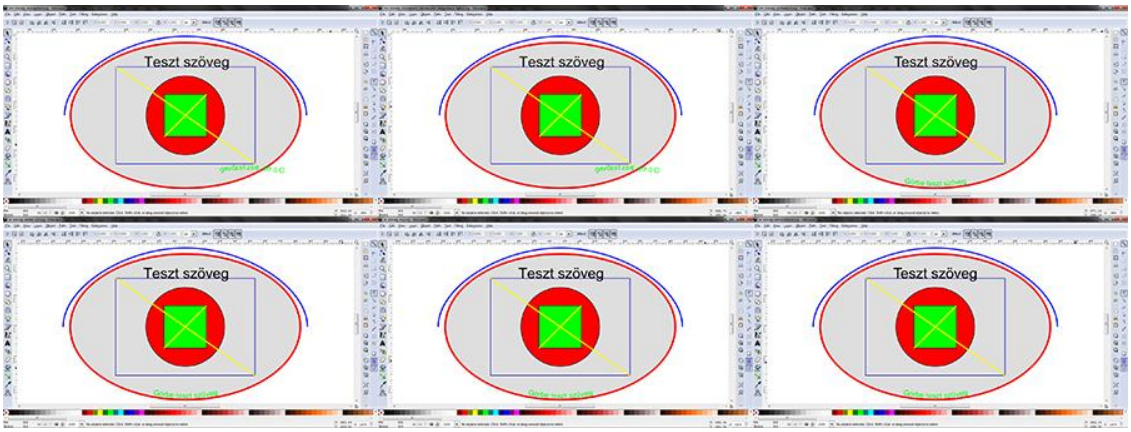
Munkánk során előfordulhat, hogy egy szoftverrel előállított grafikát, legyen az egy komplett térkép vagy csak egy piktogram, egy másik programban szeretnénk megnyitni. Erre a SVG formátum használata legtöbbször tökéletesen megfelel, bár vannak állományok, amelyek olyan beállítási opciókat alkalmazva jöttek létre, aminek következtében más programok által nehezen értelmezhető fájlszerkezettel rendelkeznek. Ennek utánajárva az eddig elkészült állományaimat megnyitottam az exportáláshoz használt szoftvertől eltérő programokban, hogy megnézzem, hogyan viselkednek egy másik környezetben.



7. ábra – CorelDraw-ban megnyitva – Felül: Adobe Illustrator (Adobe CEF, SVG, Körvonalként), Alul: Inkscape (Inkscape SVG, Plain SVG, Optimized SVG)



8. ábra – Adobe Illustratorban megnyitva – Felül: CorelDraw (Szöveg szövegként, Szöveg szövegként betűkészlet beágyazásával, Görbeként), Alul: Inkscape (Inkscape SVG, Plain SVG, Optimized SVG)



9. ábra – Inkscape-ben megnyitva – Felül: CorelDraw (Szöveg szövegként, Szöveg szövegként betűkészlet beágyazásával, Görbeként), Alul: Adobe Illustrator (Adobe CEF, SVG, Körvonalként)

A teszt során tisztán látszik, hogy megjelenésre legjobban az Adobe Illustrator által exportált SVG fájlokat kezeli a másik két szoftver. Az alakzatok egymáshoz képest megfelelően helyezkednek el és a színek is megegyeznek az előre meghatározottakkal (utóbbi a későbbiekben is igaz). CorelDraw-ban az alakzatok és a körvonalak mérete különbözik az eredetitől, ez feltételezhetően felbontásbeli eltérésekből adódhat. Inkscape esetében ezek is megegyeznek, viszont nagy hibaként róható fel ennél a programnál, hogy nem tudja megfelelően kezelni a megnyitott állományok rétegszerkezetét, egészen pontosan egy réteget sem jelenít meg, viszont a grafikát igen. Ez nem csak az Illustrator által exportált SVG fájloknál figyelhető meg, hanem minden esetben, még a saját maga által készített állományoknál is. A görbe mentén futó szövegeknél minden esetben igaz, hogy görbeként kezelik az egyes karaktereket, még

akkor is, ha szövegként lettek exportálva. A normál szövegeknél mindig a megfelelő módon jelennek meg a karakterek.

A CorelDraw-ban készített állományokat is viszonylag jól kezelik a programok, egyedül a görbe mentén futó szövegek esetében tapasztalható látványos eltérés. Ha a szövegek egyszerű szövegként voltak exportálva, akkor a görbe mentén futóak nem a megfelelő helyen, illetve Illustrator esetében eltérő méretben is jelentek meg. Ez annak tudható be, hogy a másik két szoftver valószínűleg különböző módon értelmezi a görbe szövegeket, mint a CorelDraw. Görbeként exportált szövegeknél annak megfelelően működik a szövegkezelés. A többi alakzat vizsgálatakor a két szoftver megjelenítése között különbségek fedezhetőek fel. Adobe Illustratornál a körvonal vastagságok kisebbek az eredeti értékeknél, ebből kifolyólag az alakzatok méretében is apró eltérések figyelhetőek meg (tizedes nagyságrendű). Ezzel ellentétben Inkscape-nél jelentősebb méretnövekedés jellemző, és a körvonalak vastagsága, illetve a betűméretek is nagyobbak.

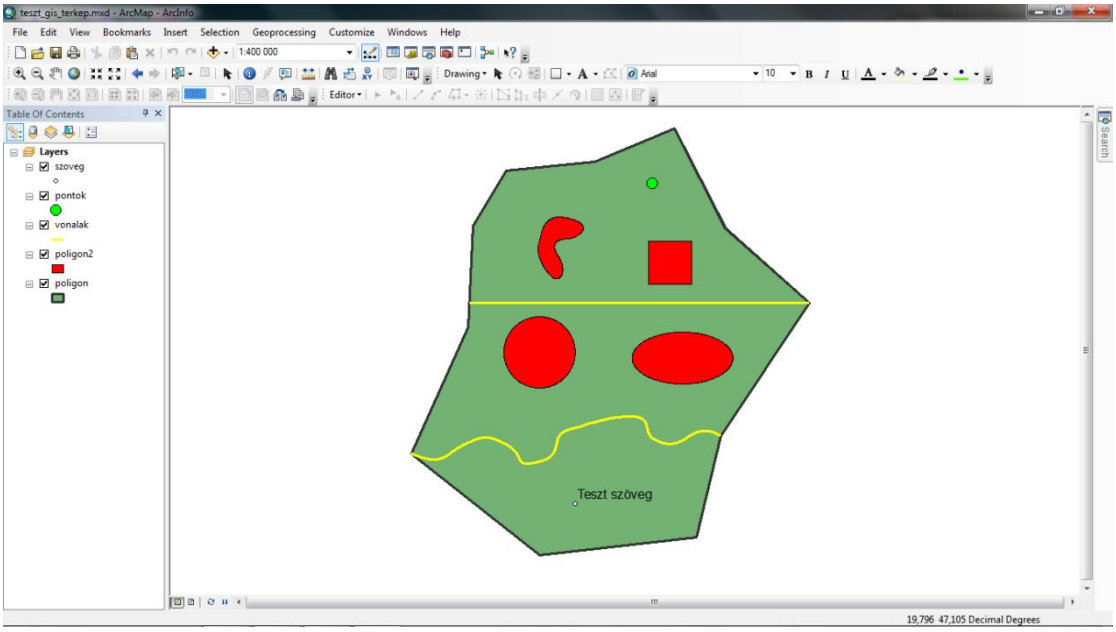
Az Inkscape-pel exportált állományok viselkedése igen ellentmondásos a különböző szoftverekben. A CorelDraw abszolút nem tudja megfelelően kezelni az ilyen fájlokat, az alakzatok rendszerint nem a helyükön jelennek meg, továbbá a szövegek mérete és megjelenése sem helyénvaló. Talán az ’optimized SVG’ mutat némi egyezést az eredeti mivoltához képest, de véleményem szerint ez nem elég erény ahhoz, hogy érdemes legyen mélyebben foglalkozni vele. Adobe Illustratorban azonban pont az ellenkezője történik. Habár a fájlok megnyitásakor a szoftver figyelmeztet, hogy a görbe mentén lévő szöveg simítva lesz, betöltés után abszolút kielégítően jeleníti meg a tartalmat. Az alakzatok mérete és a körvonalak vastagsága eltér az eredetitől, de nem olyan nagymértékben, hogy azt ne lehetne korrigálni. A szöveges elemek ebben a szoftverben a legmegfelelőbbek. A normál szöveg mind a három fájlnál jól rajzolódik ki, a görbe mentén futó szöveget pedig ugyan karakterenként importálja a program, de megnyitás után valóban szövegként értelmezi őket, ellentétben az eddigi esetekkel.

4.2. SVG a térinformatikai szoftverekben

Egy másik módja a térképi tartalommal rendelkező SVG állományok létrehozásának a térinformatikai szoftverek használata. Ennek megfelelően több ilyen típusú program

segítségével is megkíséreltem létrehozni SVG fájlokat. Ezt azért tartottam fontosnak, mert számos esetben előfordulhat, hogy valamilyen térinformatikai alapanyagból kiindulva szeretnénk készíteni térképet, viszont az ilyen szoftverek nem rendelkeznek megfelelő eszközökkel esztétikus térképek szerkesztéséhez, ezért szükség lehet egy külön ilyen célra fejlesztett szoftverben való megnyitásukra és továbbszerkesztésükre.

A teszteléshez előre elkészített shape fájlokat használtam, amiknél elsősorban az egyszerűség volt a mérvadó, nem pedig a térképi hitelesség. ArcGIS szoftverben megrajzoltam azokat a legalapvetőbb alakzatokat, amelyek egy térinformatikai programban generált vagy azzal megrajzolt térkép esetében előfordulhatnak. Külön shape fájlokat készítettem el pontszerű, vonalas, poligonos, illetve szöveges elemekhez. Utóbbi vonatkoztatási helyéül egy pontot használtam, amit elláttam egy fiktív attribútum adattal, hogy a későbbi felhasználásnál ezzel felcímkézve a szöveges elemet reprezentálja.



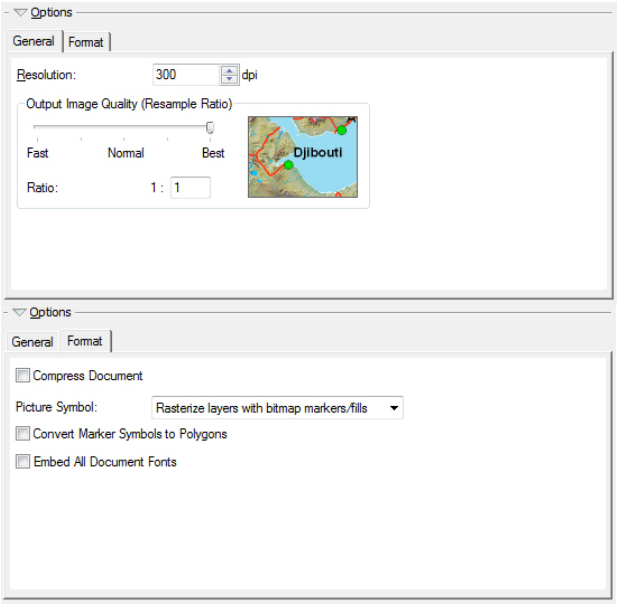
10. ábra – Az alakzatokat tartalmazó shape fájlok ArcMapben

Mindegyik alakzat típushoz meghatároztam valamilyen megjelenési beállítást, mint a kitöltési szín vagy a körvonal színe, illetve vastagsága. Ettől térképszerűbbnek tűnik a tartalom, illetve az SVG fájl exportálásánál az is kiderült, hogyan kezelik a GIS programok ezeket a tulajdonságokat. A beállításokat mindegyik szoftverben külön el kellett végeznem, mivel a megjelenési tulajdonságok helyi jellemzőnek számítanak, a shape fájlok csak magukat az alakzatokat tartalmazzák. Itt nem feltétlen törekedtem arra, hogy az értékek megegyezzenek mindegyik programban, fontosabbnak találtam az

SVG állomány exportálásának a sikerességét. Ez nem minden esetben valósult meg, mert sajnos nem mindegyik GIS szoftver támogatja alapesetben az SVG-t, és megfelelő kiegészítő modul sem mindig létezik hozzájuk.

4.2.1. Exportálás ArcGIS-szel

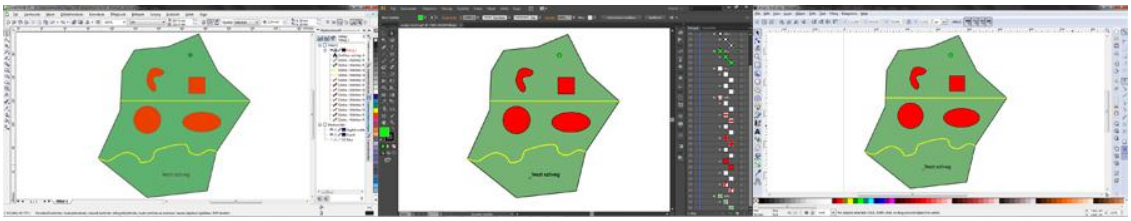
Az ArcGIS 10-es verziójával egészen egyszerűen lehet működő SVG állományokat készíteni, talán a legkönnyebben az ilyen jellegű programok közül. ArcMapen belül a ’File’ menüben található ’Export Map’ almenü választásánál, a többféle raszteres és vektoros fájlformátum között megtalálható az SVG is. Ezen felül még apróbb exportálási beállítások is elvégezhetőek, ami térinformatikai programoknál nem mindig jellemző.



11. ábra – SVG exportálási beállítások ArcMapben

A kész SVG fájl szerkezetében megfigyelhető néhány alapvető jellemző. Az egyes rétegeket a megszokott módon `<g>` elemekbe rendezi a program, azok neveit pedig azonosítóként adja meg. Minden alakzat egy `<path>` elem segítségével, vagyis görbékkel van leírva, annak ellenére, hogy legtöbbjüknek, mint például a körnek vagy az ellipszisnek, van előre definiált eleme a nyelvben. A teszt szöveget reprezentáló pont és maga a szöveg külön rétegen helyezkedik el, előbbi azonosítója a shape fájl nevéből ered (’szoveg’), utóbbié egy automatikusan generált kifejezés (’Labels’). A címkeként

funkcionáló szöveg az egyedüli elem (a szabálytalan sokszögön kívül), ami a kifejezetten hozzá tartozó `<text>` elem segítségével van meghatározva. A stílusokat az egyes tulajdonságok attribútumként való használatával definiálja a program.

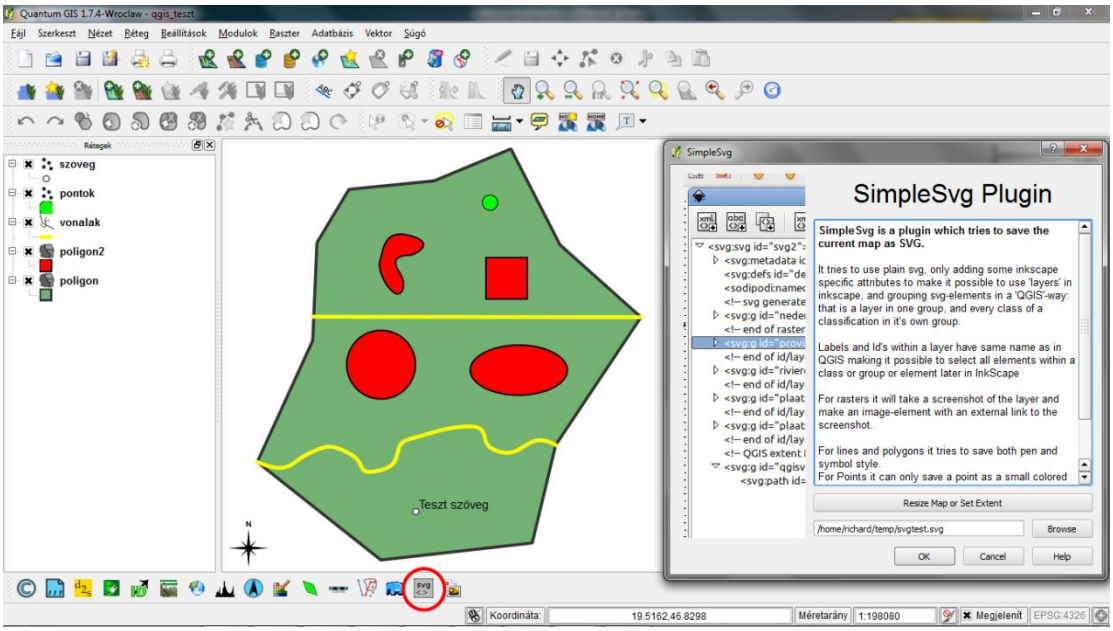


12. ábra – ArcMap által generált SVG fájl megnyitva grafikai programokban (CorelDraw, Adobe Illustrator, Inkscape)

Grafikai programokban megnyitva a fájlt igen jó eredményt kapunk. Az indokolatlanul sok csoport szétbontása, az ArcGIS által generált maszkrétegek törlése, illetve CorelDraw-ban az objektumok zárolásának feloldása után teljesen kezelhető formává válik a munkánk. Az egyetlen érdekesség, hogy a vörös alakzatok kitöltését és körvonalát külön objektumként kezelik a programok. Ez természetesen nem a színük miatt van, hanem egyszerűen exportálási vagy olvasási (értelmezési) hibaként jöttek létre. Ez a hiba azonban könnyen korrigálható és megjelenésében nem is befolyásolja a grafikát.

4.2.2. Exportálás Quantum GIS-szel

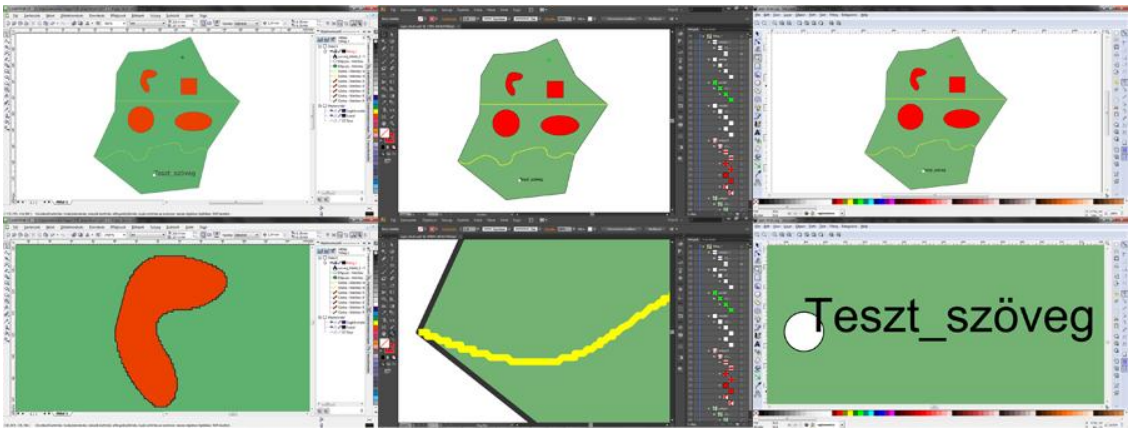
A Quantum GIS 1.7.4.-es (Wrocław) verziója nem tartalmaz olyan funkciót, amellyel elő lehet állítani SVG fájlt, viszont letölthető hozzá egy kiegészítő modul, amivel remekül elvégezhető ez a művelet. Ennek a SimpleSvg elnevezésű modulnak a letöltése után, a funkciók között megjelenik egy ikon, amire kattintva egy exportáló ablak ugrik fel, amiben megadható az exportálás pontos helye. [11]



13. ábra – SVG állomány exportálása Quantum GIS-ben, SimpleSvg modul használatával

A kiegészítő modul leírásában a fejlesztők garantálják, hogy az ilyen módon létrehozott SVG állományok jól kezelhetőek és szerkeszthetőek lesznek Inkscape grafikai programban. Exportálás előtt a megjelenési beállításokat ebben az esetben is elvégeztem az ArcMappel nagyrészt megegyező módon.

Az exportált fájl forráskódját vizsgálva valóban észrevehetőek hasonlóságok az Inkscape által generált SVG fájlokhoz képest. Amellett, hogy a program a rétegeket itt is `<g>` elemekbe rendezi, valamint a stílusmegadásnál magát a tulajdonságot használja attribútumként, az Inkscape saját névterébe tartozó ’`inkscape: …`’ attribútumok is felfedezhetőek benne. A poligonok és a vonalak megrajzolása `<path>` elemekkel, a pontokhoz tartozó kör alakú jelek megalkotása a `<circle>` elemmel történik. A szövegként funkcionáló címke és a hozzá tartozó pont elem külön, a ’szoveg’ és a ’szoveg_labels’ azonosítójú rétegeken találhatóak meg.



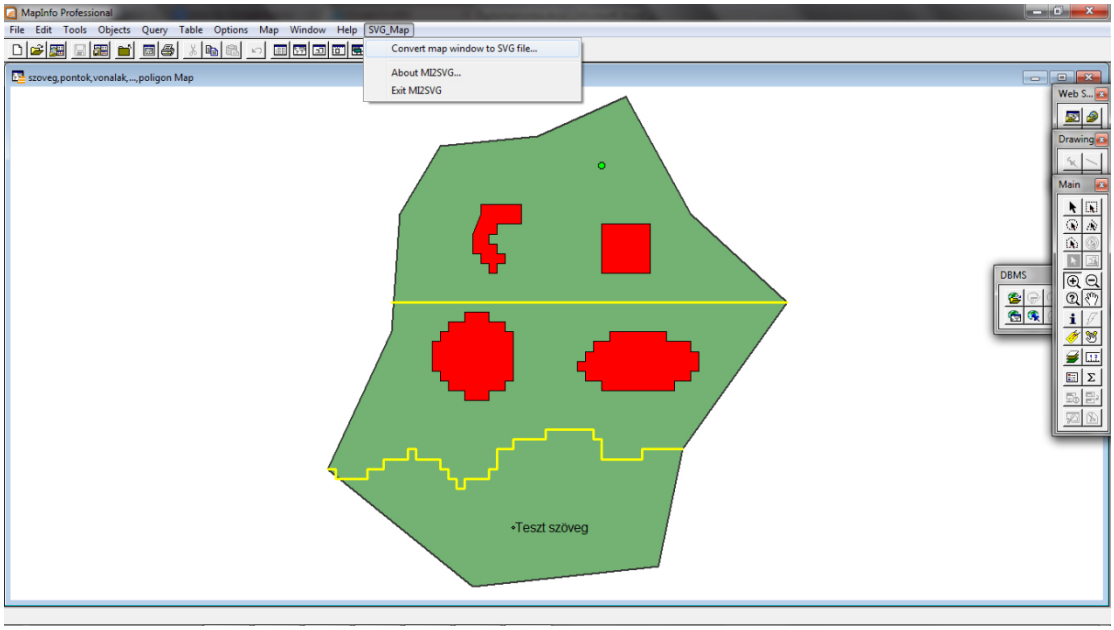
14. ábra – Quantum GIS által generált SVG fájl megnyitva grafikai programokban (CorelDraw, Adobe Illustrator, Inkscape), illetve a megjelenítési hibák

A különféle grafikai szoftverek megfelelően kezelik az állományt, azonban némi hiba fedezhető fel annak elemzésekor. A `<path>` elemek által kirajzolt görbéknél és görbével határolt alakzatoknál (pl. kör, ellipszis) a vonal futása nem kisimult, hanem barázdás. Ez valószínűleg exportáláskor, a koordináták nem megfelelő meghatározásából eredhet. A zavaró zárolások és csoportosítások eltávolítására jelen esetben is szükség van, illetve a Quantum GIS generál egy vörös keretet is a grafikának, amit szintén érdemes törölni. Szöveges elemeknél a szóköz helyére alulvonás kerül, a betűméret pedig nagyobb CorelDraw-ban, mint a másik két szoftverben. A pontszerű elemeket, ahogyan a szöveghez tartozó pontot is, megfelelően jelenítik meg a programok, hiszen ezek `<circle>` elemként voltak meghatározva.

4.2.3. Exportálás MapInfoval

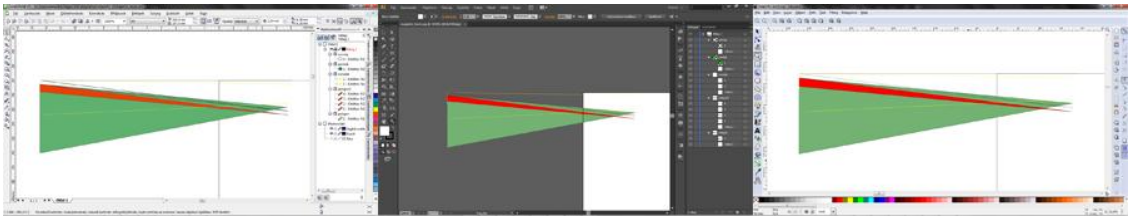
MapInfo 10-ben, a Quantum GIS-hez hasonlóan, nincsen alapvető funkció SVG állományok létrehozására. Létezik azonban két kisebb program, amik használatával elméletileg lehetőség nyílik ennek a feladatnak az elvégzésére, de gyakorlatban már nem működtek megfelelően. Az egyik ilyen programnál, a DBx GEOMATICS által fejlesztett SVGMapMakernél, az volt a legnagyobb probléma, hogy csak az ingyenes, demó változatának a kipróbálására volt lehetőség, ami exportálást nem engedélyez. A másik programmal, a 2004-ben fejlesztett MI2SVG-vel, már eredményesebb volt a próbálkozás. A MapInfo megnyitása után el kell indítani egy MI2SVG.MBX fájlt, ami után megjelenik egy ’SVG_Map’ elnevezésű menüpont a programban, aminek a

’Convert map window to SVG file…’ almenüjével van lehetőség elkészíteni az SVG állományt. Ez a funkció csak akkor működik, ha a térkép ablak nincsen minimalizálva vagy maximalizálva.



15. ábra – SVG állomány exportálása MapInfoban, MI2SVG program használatával

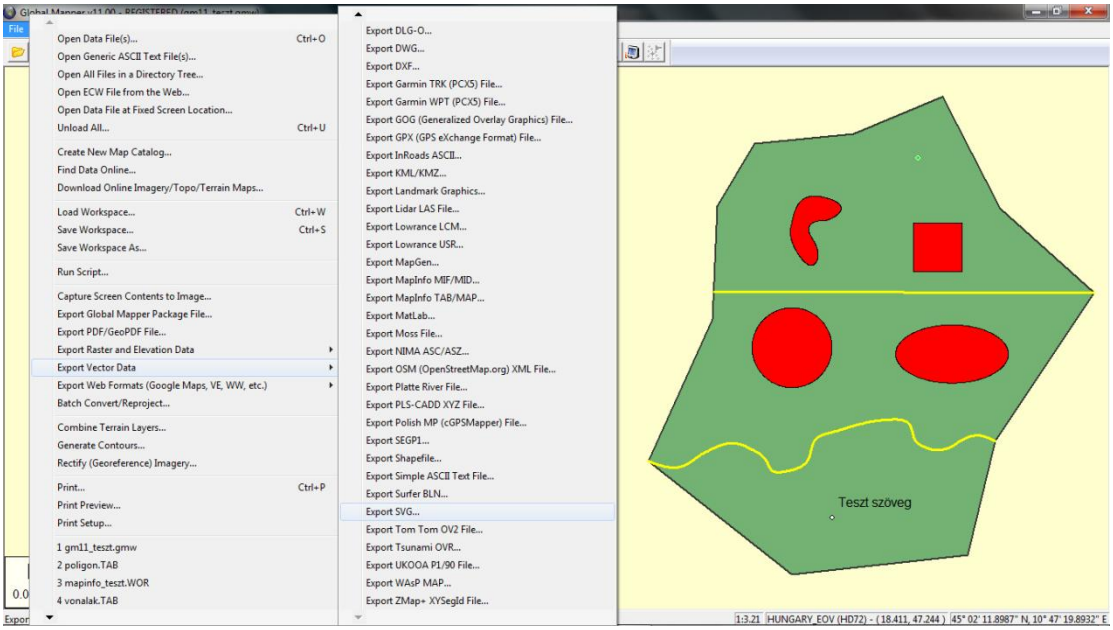
Az így elkészült fájl felépítése az előírásoknak tulajdonképpen megfelel, a rétegek `<g>` elemekbe vannak sorolva, azok nevei azonosítóként szerepelnek, a stílusok pedig a `style` attribútum segítségével vannak meghatározva. Van azonban egy hiba, ami miatt az SVG által kirajzolt grafika végeredménye egyáltalán nem hasonlít az eredeti tartalomhoz. A `<path>` elemmel létrehozott alakzatok koordinátáinak megadásakor valamiért a kezdőpontok kivételével mindenhova nullákat generál a program, ebből fakadóan a kirajzolt grafika teljesen torzult képet mutat a grafikai szoftverekben megnyitva. Emellett exportáláskor a címkeként használt szöveges elem sem került bele a fájlba. Ezeket a hibákat figyelembe véve gyakorlatilag lehetetlen utómunkálatokat végezni az állományon, tehát a további vizsgálatuknak sincs értelme.



16. ábra – MI2SVG program segítségével MapInfo-ban generált SVG fájl megnyitva grafikai programokban (CorelDraw, Adobe Illustrator, Inkscape)

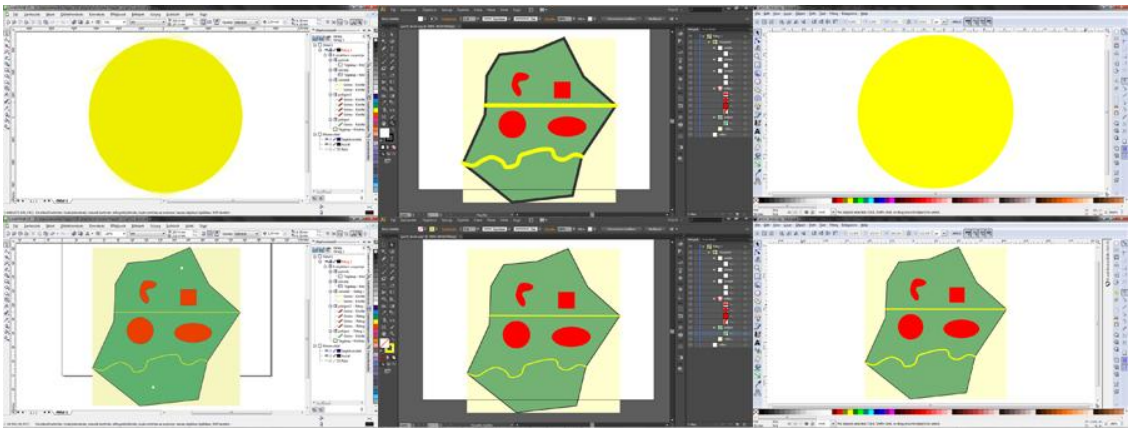
4.2.4. Exportálás Global Mapperrel

Global Mapper 11-es szoftvert használva, az ArcGIS-hez hasonlóan, könnyedén létre lehet hozni SVG fájlokat, ugyanis az alapvető exportálási lehetőségek között itt is szerepel az SVG. Az opció választásakor, a Global Mapperben megszokott módon, a felugró exportálási ablakban be lehet állítani az exportálni kívánt tartalmat határoló képzeletbeli négyszög északnyugati és délkeleti sarokkoordinátáit is.



17. ábra – SVG állomány exportálása Global Mapperben

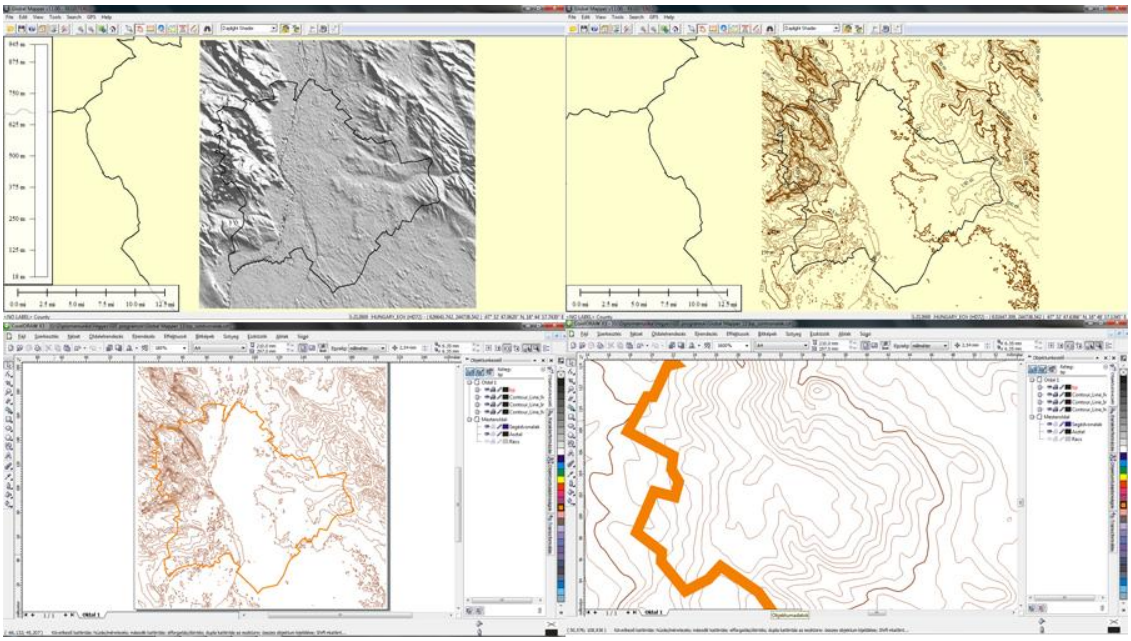
Az SVG fájl legelején a program definiál két mintázatot `<pattern>` elemek segítségével, amiket a pontok kitöltéséhez használ fel, habár a grafika ezt nem igényli. A továbbiakban a megszokott módon épül fel az állomány. A rétegek neve azonosítóként jelenik meg a `<g>` elemekben, a stílusok pedig `style` attribútumokban vannak meghatározva. Bizonyos objektumok körvonal vastagságát nullának, másokét pedig eredeti értékük többszörösének határozza meg a program. A poligonok `<path>`, a vonalak `<polyline>`, a pontok érdekes módon `<rect>` elemekkel definiáltak. A szöveges elem itt sem szerepel az állományban.



18. ábra – Felül: Global Mapper által exportált SVG fájl grafikai programokban megnyitva (CorelDraw, Adobe Illustrator, Inkscape), Alul: javított megjelenés

Grafikai programokban megnyitva a fájlt, a nem megfelelő körvonalvastagság kezelési mód rögtön szembetűnik, hiszen komoly eltéréseket eredményez a rajz képében. CorelDraw és Inkscape esetében a vonalaknak és a háttérben levő sokszög poligonnak a körvonala olyan vastag, hogy kitakarja az egész grafikát. Ezzel szemben a vörös színű poligonoknak a körvonala eltűnik, mivel a vastagságuk nullára lett állítva. Adobe Illustratorban már nem ennyire szélsőséges a megjelenés. Az eltérések valószínűleg azért jöhetnek létre, mert a forráskódban nem szerepel mértékegység a vastagságok megadásakor, ezért a programok különbözően értelmezhetik az értékeket. Ezek a hibák kisebb alakítások után persze helyrehozhatóak, de a pontokhoz rendelt mintákat nem tudják megfelelően kezelni a grafikai szoftverek, illetve a pontok valójában négyzetek, hiszen a Global Mapper így exportálta őket.

Egy másik hasznos tulajdonsága a Global Mappernek, hogy egyszerűen lehet szintvonalakat generálni a segítségével, amik SVG állományként is exportálhatóak, majd felhasználhatóak bármilyen grafikai szoftverben, például egy térkép domborzatrajzának alapjaként. Ezeket a magassági adatokat az ingyenesen elérhető SRTM (Shuttle Radar Topography Mission) domborzatmodellből tölti le a program. A meghatározott területre generált domborzatrajz eredetileg valamilyen magassági színezéssel van ábrázolva, azonban a szoftver funkcióival könnyen készíthető belőle megfelelő magassági szinteket bemutató szintvonalrajz.



19. ábra – SVG szintvonalrajz generálása Global Mapperben, majd annak megnyitása CorelDraw-ban

A fenti példában Budapest és környékéről generáltam 25 méteres alapszintközzel szintvonalakat, amik közül minden ötödiket főszintvonalként, vastagítva ábrázoltam. Ezután SVG formátumban mentettem az állományt, majd megnyitottam azt CorelDraw-ban. Az eredmény abszolút kielégítő, a generált szintvonalak szabadon alakíthatóak, adott esetben hipszometrikus ábrázoláshoz is kitűnően használhatóak. Közelebbről vizsgálva látszik, hogy még a görbék futása is megfelelő. Nagyobb terület, például az egész Kárpát-medence esetében már körülményesebb végrehajtani ezt a műveletet, mert a nagyszámú szintvonal generálása több időbe telik a Global Mappernek, valamint a grafikai programok is nehezebben kezelik az állományt.

4.3. SVG a weben

SVG fájlok használata a világhálón manapság majdnem olyan fontos, mint egy grafikai programban. Ez a raszteres képekkel szembeni számos előnyükből fakadóan alakult így ki. Az egyik ok, hogy az ilyen állományok szöveg alapúak, ami azért hasznos az őket tartalmazó honlapok számára, mert a keresőmotorok így beleláthatnak a forráskódjukba kulcsszavak után kutatva, ezáltal a weblap könnyebben megtalálhatóbbá válhat. Vektoros mivoltukból adódóan, a grafika szabadon méretezhető, nem jönnek létre torzulások a méret növelésével. A modern honlapoknál ez kiemelkedően fontos, hiszen

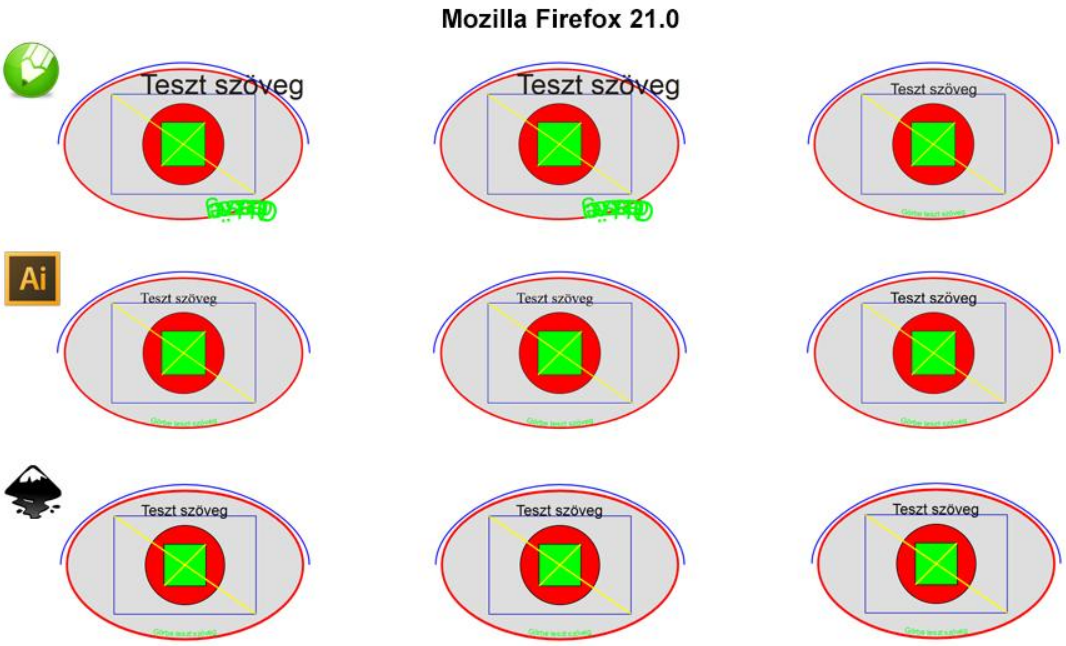
nem tudható előre, hogy milyen felületről lesz megtekintve egy adott oldal (különböző felbontású monitorok, táblagépek, okostelefonok), ezért a honlapkészítők igyekeznek felbontás független weblapokat létrehozni. Programozási nyelv (JavaScript) segítségével manipulálhatóak, ezáltal például interaktív térképek is készíthetőek belőlük webes felületen. Nem utolsó sorban a fájlok mérete a legtöbb esetben kisebb, mint a raszteres állományoké, ami a honlapok betöltési idejét rövidítheti, illetve kevesebb helyet foglalnak a webtárhelyből is.

A hivatalos W3C dokumentáció szerint jelenleg az SVG-t az összes nagyobb böngésző támogatja. Internet Explorernél csak a 9-es verzió óta élvez támogatást, a 8-as vagy annál régebbi változatoknál szükség van egy plug-in telepítésére, ami segít a megjelenítésben. Ilyen kiegészítő modul például az Adobe által fejlesztett, ingyenesen letölthető Adobe SVG Viewer.

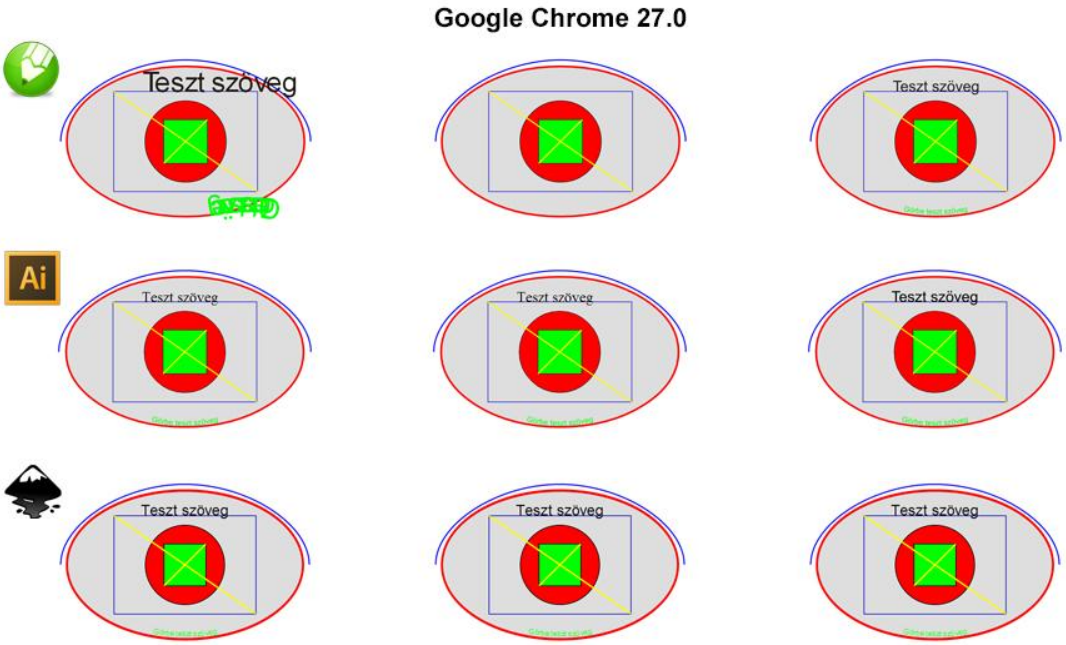
SVG tartalmakra kétféleképpen lehet hivatkozni egy honlap keretein belül. Az egyik lehetőség, hogy közvetlenül az oldal forráskódjába írjuk az SVG kódot egy nyitó `<svg>` és egy záró `</svg>` elem közé. A másik módszer, hogy egy külső *.svg* kiterjesztésű fájlt ágyazunk be a HTML kódba az `<embed>`, az `<object>`, vagy az `<iframe>` elemek segítségével, aminek a tartalma természetesen ugyanúgy megjelenik a böngészőben, mint az első lehetőségnél:

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="50" y="20" width="50" height="50"
  style="fill:blue; stroke:red; stroke-width:3;"/>
</svg>

<embed src="kulso.svg" type="image/svg+xml"></embed>

<object data="kulso.svg" type="image/svg+xml"></object>

<iframe src="kulso.svg"></iframe>
```

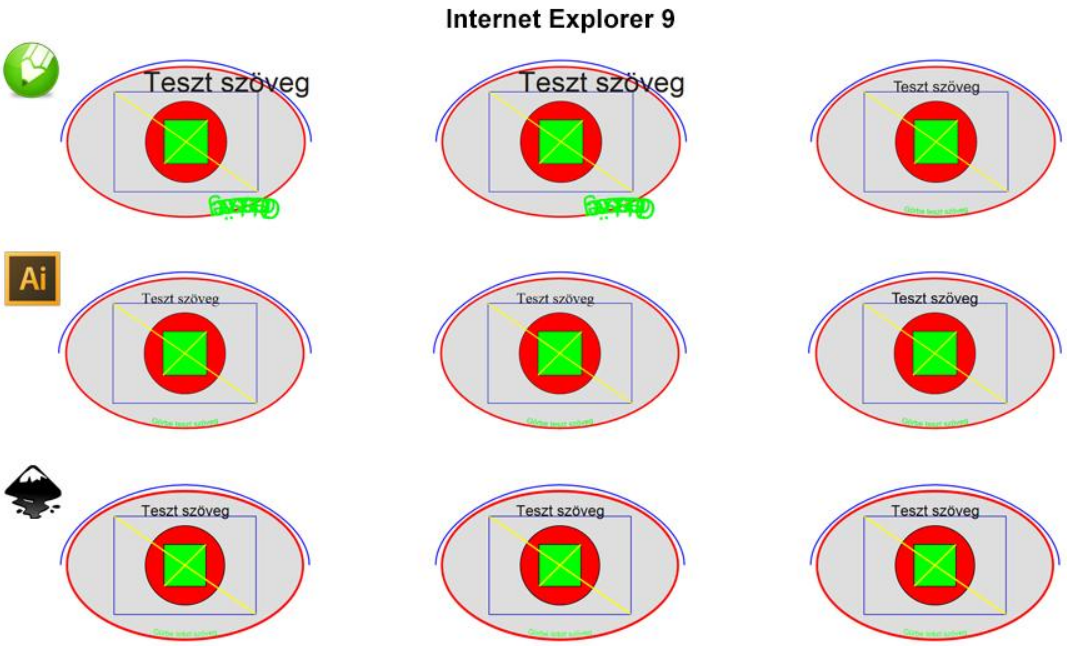
A következő példákban három fő böngészőben, a Mozilla Firefoxban, a Google Chrome-ban, és az Internet Explorerben megnyitva láthatóak a korábban CorelDraw, Adobe Illustrator, illetve Inkscape által elkészített SVG állományok. Az első két oszlopban mindig azok a fájlok kerültek, amikben szövegként lettek exportálva a szöveges elemek, a harmadik oszlopban pedig a görbeként exportált szöveges elemeket tartalmazóak láthatóak.



20. ábra – Grafikai szoftverek által készített SVG állományok Mozilla Firefoxban megnyitva



21. ábra – Grafikai szoftverek által készített SVG állományok Google Chrome-ban megnyitva



22. ábra – Grafikai szoftverek által készített SVG állományok Internet Explorer 9-ben megnyitva

Jól látszik, hogy a böngészők majdnem ugyanúgy kezelik az egyes állományokat, az alakzatok mindenhol megegyeznek és a szövegeknél is csupán egy kisebb eltérés vehető észre a Google Chrome-ban megnyitott második CorelDraw állománynál. Ennél az esetnél a szöveges elemek szövegként, a betűkészlet beágyazásával lettek exportálva és ezt a böngésző nem tudja megfelelően értelmezni, ezért nem is jeleníti meg.

A legjobb eredmény a görbeként exportált szövegeknél, illetve az InkScape által generált SVG fájloknál született. Ezeknél az állományoknál mindig megegyező módon jelentek meg a grafikák, annak ellenére, hogy a görbeként értelmezett szövegek három eltérő szoftverrel lettek elkészítve. Adobe Illustratornál és CorelDraw-nál a szöveges elemekben láthatóak kisebb-nagyobb eltérések azokban az esetekben, amikor szövegként lettek exportálva.

Az Internet Explorer 8-as verziója nem képes megjeleníteni SVG fájlokat, viszont az Adobe SVG Viewer telepítése után már ugyanazokat az eredményeket kaptam, mint a 9-es változatnál, tehát kijelenthető, hogy a korábbi verziók használatakor is kiküszöbölhető a megjelenítési probléma.

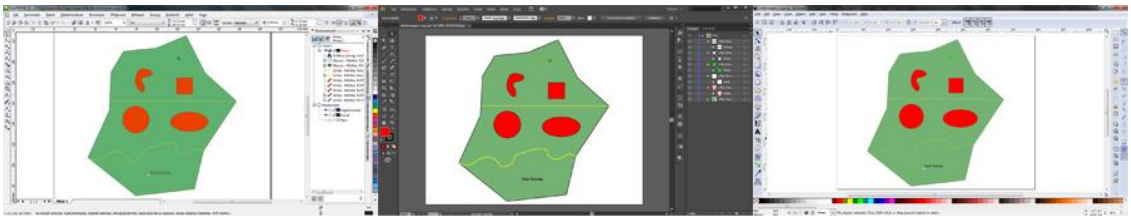
4.3.1. SVG fájl készítése Indiemapperrel

Az Indiemapper egy kiváló internetes térképes alkalmazás, amellyel rendkívül egyszerűen lehet térképeket készíteni különböző földrajzi adatokból. Előnye, hogy legalább olyan hatékonysággal lehet vele térinformatikai adatokból SVG állományokat létrehozni, mint egy direkt erre a célra létrehozott programmal, emellett még ingyenes is a szolgáltatás és feltelepítésre sincs szükség.

Alapanyagként felhasználhatóak *.kml*, *.shp*, és *.gpx* kiterjesztésű állományok, illetve shape fájlok használatakor a hozzájuk tartozó attribútum adatokat tartalmazó DBF állományok is. Az objektumok megjelenése ugyanolyan módon változtatható, mint a GIS szoftverekben, illetve lehetőség van különböző vetületek között is válogatni. Az elkészült térképeket el lehet menteni az Indiemapper saját IMP formátumában, valamint *.jpg*, *.png*., és *.svg* kiterjesztésben is.

23. ábra – Megnyitott shape fájlok Indiemapperben

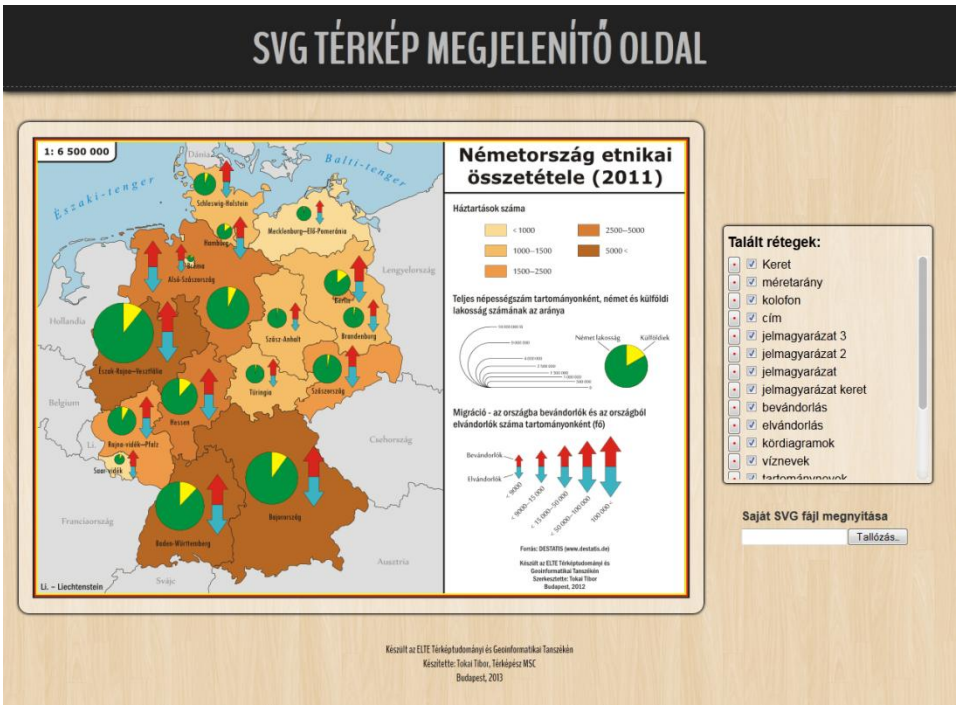
Az így elkészült SVG fájlok szerkezete a legoptimálisabb módon jön létre, az alapvető elemek mellett az Inkscape névterébe tartozóak is szerepelnek benne. Természetesen ebben az esetben is keletkeznek fölösleges, illetve zárolt elemek grafikai szoftverben megnyitva a fájlt, de eltávolításuk után tökéletesen kezelhetővé és szabadon alakíthatóvá válik az állomány.



24. ábra – Indiemapperben készített SVG fájl megnyitva grafikai programokban (CorelDraw, Adobe Illustrator, Inkscape)

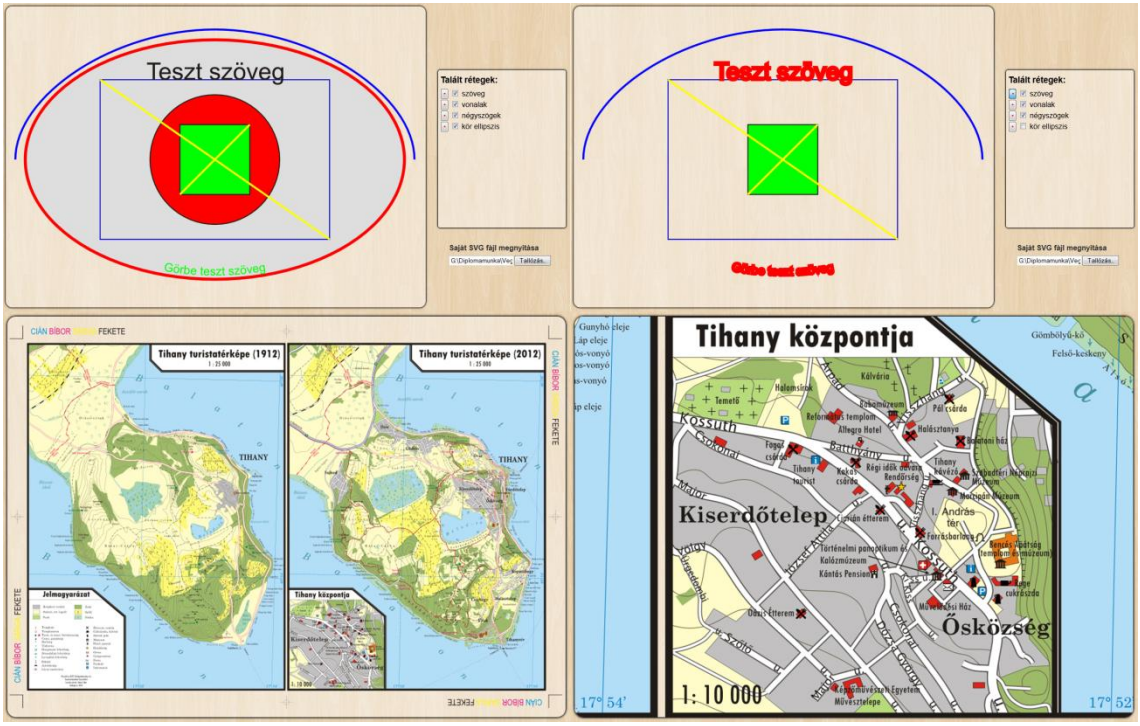
5. A szemléltető honlap bemutatása

Dr. Gede Mátyás korábbi munkáját [12] felhasználva elkészítettem egy szemléltető oldalt SVG fájlok webes megjelenítésére. A honlap elsődleges célja az volt, hogy egy letisztult, de modern megjelenésű felhasználói interfészen keresztül bárkinek lehetőséget nyújtson interneten megtekinteni saját állományait. A grafika egy főablakban jelenik meg, ami mellett megtalálható egy rétegkezelési, valamint egy fájl megnyitáshoz szolgáló felület is.



25. ábra – Az SVG térkép megjelenítő oldal

Az SVG állomány tartalmát megjelenítő részben a kép az egér segítségével szabadon méretezhető, illetve pozícionálható. A rétegkezelői felületen a grafika elkészítésekor létrehozott rétegek szerepelnek, amelyek ki-be kapcsolhatóak, valamint egy másik gomb használatával kiemelhető a tartalmuk. Ez a funkció szintén megkönnyítheti az állomány vizsgálatát. A felhasználó saját merevlemezén található fájljának megnyitása után az új tartalom megjelenik a főablakban, annak rétegei pedig a rétegkezelői felületen.



26. ábra – Saját SVG fájl megnyitása, felnagyítása és a rétegkezelési funkciók

A fenti ábrán látható, hogy az állományok megnyitása után milyen műveletek végezhetőek el rajtuk. Látszik, hogy a kört és az ellipszist tartalmazó réteg kikapcsolásával, illetve a szöveges elemek kiemelésével hogyan fest a grafika, vagy a térkép esetében a tartalom milyen szinten nagyítható.

A honlap HTML és JavaScript nyelvek segítségével íródott, egyes részei igen összetett forráskóddal rendelkeznek. Betöltésekor a főablak nem üresen jelenik meg, hanem egy alap SVG térképet tartalmaz. Erre azért van szükség, hogy az oldal akkor is kipróbálható legyen, ha valaki nem rendelkezik SVG fájlal. Az állományra hivatkozás egy `<object>` elemmel történik, aminek az `onload` attribútuma egy `init()` függvényt is meghív.

```
<object id="svgobj" width="920px" height="660px"
data="Nemetorszag_nepesseg.svg" type="image/svg+xml" onload="init()">
</object>
```

Ez a függvény felelős többek között a rétegkezelői felület feltöltéséért, valamint a főablakban történő egérműveletek működéséért is (mozgatásra és görgetésre történő folyamatok). Megkeresi az SVG állomány első `<svg>` elemét, ami után egy ciklus segítségével felkutatja az összes `<g>` elem `id` attribútumának tartalmát. Ezek az elemek képviselik az egyes rétegeket, az azonosítójuk pedig azok neveit. Ezután egy másik

ciklussal létrehozza a rétegkezelői felületen belül a rétegeket és a hozzájuk tartozó kapcsolókat. Ez a gyakorlatban úgy megy végbe, hogy a `layers` azonosítójú elemen belül készíti el a kapcsolókhoz tartozó `<input>` elemeket, amik után az azonosítóból származó réteg nevét írja.

```
svgObj=document.getElementById('svgobj'); // az <object> elem
svgDoc=svgObj.contentDocument.getElementsByTagName('svg')[0];
layersDiv=document.getElementById('layers');
var id,scrollEvent;
var layers=[];

for(var i=0;i<svgDoc.childNodes.length;i++)
  if (svgDoc.childNodes[i].tagName=="g")
  layers.push(svgDoc.childNodes[i].id);

for(i=layers.length-1;i>=0;i--)
  {
        id=layers[i];
        layersDiv.innerHTML+='<input type="button" value="&bull;"
style="color:red; width:10px;"
onmousedown="highlightLayer(\''+id+'\')"
onmouseup="unHighlightLayer(\''+id+'\')" /> ';
        layersDiv.innerHTML+='<input type="checkbox" checked
onclick="setLayer(\''+id+'\',this.checked)" />
'+nameDecode(id)+'<br/>';
  }
```

Egy másik SVG fájl megnyitásához a `readFile()` nevű függvényre van szükség, ami a feltöltést szolgáló `<input>` elem `onchange` attribútumával van meghívva. Ez a függvény meghív egy másik, `show()` nevű függvényt is, ami a beolvasott SVG fájl tartalmát base64 kódolás segítségével a HTML számára is érthetővé alakítja. Ez egy tartalomkódolási forma, amely bináris, illetve speciális karakterekből álló adatokból ASCII karaktersorozatot állít elő, vagyis például képeket képes szöveges formában leírni. Ez a tartalom ezután az `<object>` elem `data` attribútumába kerül, ami később kirajzolja a kívánt grafikát.

```
function readFile()
  {
    var reader=new FileReader();
    reader.onload=function(file) { f=file.target.result; show(); };
    reader.readAsText(document.getElementById("txtfile").files[0])
  }
```

```
function show()
  {
    var b64=btoa(unescape(encodeURIComponent(f)));
    document.getElementById('svgobj').setAttribute
("data",""+b64);
  }
```

A honlap az alábbi webcímen (URL) található:

http://mercator.elte.hu/~totqaat/svg/

6. Összefoglalás

Munkám során igyekeztem mélyre menően bemutatni az SVG állomány igazi erejét, vagyis azt, hogy minden a térképészet által ténylegesen használt felületen megfelelően megállja a helyét, legyen az valamilyen szoftver vagy egy weblap a világhálón. Ehhez már a dolgozatom elején szükségesnek tartottam tisztázni, hogy mi is a lényeges különbség a raszteres és a vektoros formátumok között, illetve az utóbbit képviselő SVG állományok általános szerkezetére is kitértem a nyelv könnyebb megértéséhez.

A továbbiakban részletesen elemeztem teszteléseim eredményeit. Grafikai szoftvereknél a legtöbb esetben kielégítő eredményt kaptam a különböző programokban megnyitott SVG fájlok tartalmának megjelenítésekor. Persze ez nem jelenti azt, hogy bármilyen térképi tartalom hibátlanul átvihető az egyik grafikai szoftverből a másikba, de véleményem szerint erre ritkábban van szükség. Ellenben hasznosabb lehet a térképi jelkulcs bizonyos részeinek (pontszerű, vonalas, vagy felületi jelek) átvitele, ami könnyedén megoldható az SVG segítségével.

Pozitív tapasztalatokkal gazdagodtam a térinformatikai szoftverek által készített állományok grafikai programokban való megnyitásakor. Az ilyen módon készített térképek nem mindig jelentek meg teljes mértékben megfelelően a grafikai szoftverekben, viszont azok eszközeivel szinte tökéletesen alakíthatóak voltak, ami azt jelenti, hogy egy adatbázisból generált összetettebb térképből egész egyszerűen lehet akár rövid idő alatt is esztétikus térképeket szerkeszteni.

Végezetül különböző böngészőkben is teszteltem az állományokat, illetve bemutattam szemléltető honlapomat, ami az interneten való megtekintésüket teszi praktikusabbá. Örömömet leltem abban, hogy az SVG honlapokon való alkalmazását jobban megismerhettem, mert úgy vélem, hogy valóban hasznos szerepük van ezen a területen is.

7. Köszönetnyilvánítás

Ezúton szeretném megköszönni témavezetőmnek, Gede Mátyásnak, a jó tanácsokat, amikkel ellátott munkám során, valamint a weblap elkészítéséhez nyújtott segítségét. Továbbá hálával tartozom családomnak, amiért mindvégig támogattak egyetemi éveim során.

8. Irodalomjegyzék

Nyomtatott irodalom

- **Elek István, 2006:** Bevezetés a geoinformatikába, ELTE Eötvös Kiadó, Budapest

- **Dailey, David; Frost, Jon; Strazzullo, Domenico; 2012:** Building Web Applications with SVG, O'Reilly Media, California

- **Eisenberg, J. David; 2002:** SVG Essentials, O'Reilly Media, California

Internetes hivatkozások

- [1] – **Wikipedia:** SVG
  http://hu.wikipedia.org/wiki/SVG

- [2] – **Wikipedia:** Vektorgrafika
  http://hu.wikipedia.org/wiki/Vektorgrafika

- [3] – **Wikipedia:** Rasztergrafika
  http://hu.wikipedia.org/wiki/Rasztergrafika

- [4] – **Wikipedia:** Comparison of vector graphics editors
  http://en.wikipedia.org/wiki/Comparison_of_vector_graphics_editors

- [5] – **Wikipedia:** Adobe Illustrator
  http://en.wikipedia.org/wiki/Adobe_Illustrator

- [6] – **Wikipedia:** CorelDRAW
  http://en.wikipedia.org/wiki/CorelDRAW

- [7] – **Inkscape Wiki:** Frequently asked questions
  http://wiki.inkscape.org/wiki/index.php/FAQ

- [8] – **w3schools:** SVG Tutorial
  http://www.w3schools.com/svg/

- [9] – **Peter's website:** SVG Tutorial
  http://www.petercollingridge.co.uk/svg-tutorial

- [10] – **SVG ELTE:** SVG Tananyag
  http://svg.elte.hu/

- [11] – **duif.net:** Simple SVG Qgis Plugin
  http://www.duif.net/qgis/simplesvg/

- [12] – **Gede Mátyás:** CorelDraw által exportált SVG fájl megjelenítése nagyítással, rétegszerkezettel
  http://mercator.elte.hu/~saman/hu/okt/svg/corel_svg_display.html

9. Nyilatkozat

Alulírott, …….. nyilatkozom, hogy jelen dolgozatom teljes egészében saját, önálló szellemi termékem. A dolgozatot sem részben, sem egészében semmilyen más felsőfokú oktatási vagy egyéb intézménybe nem nyújtottam be. A diplomamunkámban felhasznált, szerzői joggal védett anyagokra vonatkozó engedély a mellékletben megtalálható.

A témavezető által benyújtásra elfogadott diplomamunka PDF formátumban való elektronikus publikálásához a tanszéki honlapon

HOZZÁJÁRULOK NEM JÁRULOK HOZZÁ

Budapest, 2013. június 7.

……
(a hallgató aláírása)