EÖTVÖS LORÁND TUDOMÁNYEGYETEM
INFORMATIKAI KAR
TÉRKÉPTUDOMÁNYI ÉS GEOINFORMATIKAI TANSZÉK

Karancs webes turista-útikalauz

Varga Balázs
térképész mesterszakos hallgató

Témavezető:
Gede Mátyás
adjunktus

ELTE Térképtudományi és Geoinformatikai Tanszék

Budapest, 2014

Tartalomjegyzék

1. Bevezetés

A diplomadolgozat elkészítéskor egy mindenki által hozzáférhető, webes útvonaltervező elkészítésére törekedtem, amely alapvető információkkal szolgál a Karancs hegység területén túrázni kívánóknak. Így bemutatja a Karancs turistaútjait, kinyerhetők vele az egyes túraszakaszok adatai és megbecsülhető a szintkülönbség.

A diplomadolgozat a korábbi szakdolgozatom anyagára támaszkodik, de új formában tárgyalja azt [Varga, 2011]. A korábbi szakdolgozat keretében elkészített felmérés anyaga kiegészül a Karancs-hegység szlovákiai részén lévő turistautakkal, valamint implementálásra kerül egy weboldal. A honlap egy beépülő webes térképi modult tartalmaz egyszerű, útvonaltervezést segítő funkciókkal, illetve letölthetők róla a túraútvonalak.

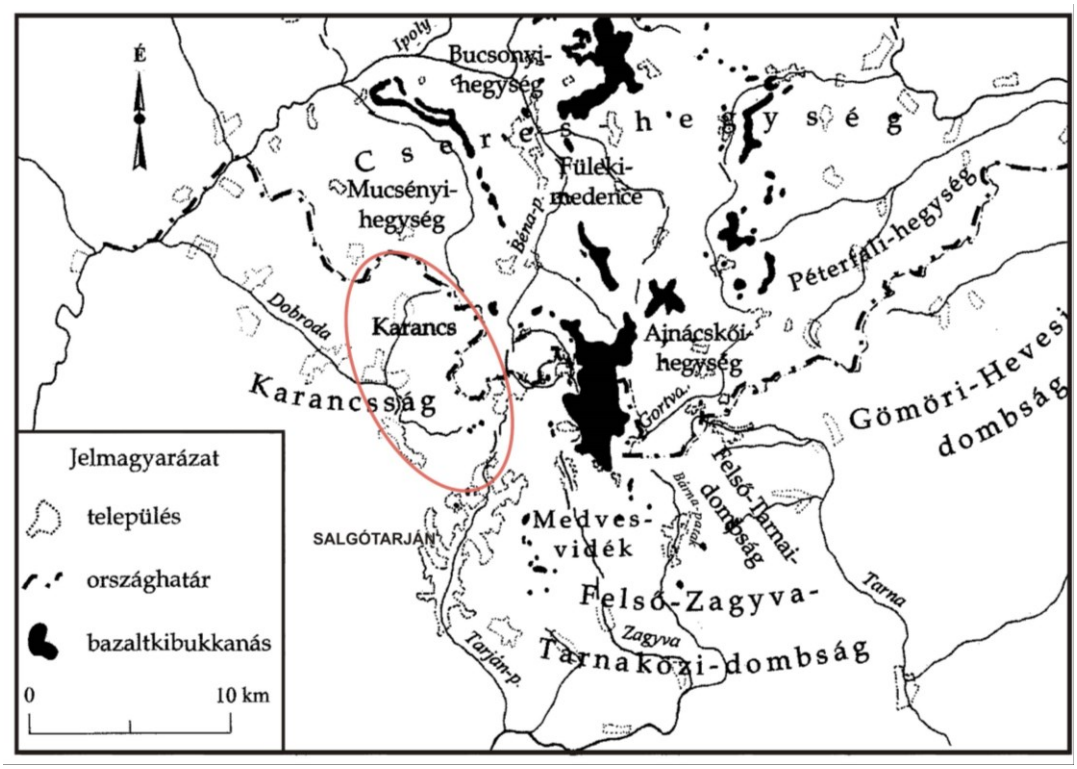
A dolgozat a munkafolyamatot tárgyalja a felméréstől a webes közzétételig, valamint kitér a felhasznált technológiára.

2. Adatgyűjtés

2.1 Terepi bejárás

2010-ben, 2011-ben és 2014-ben végeztem a karancsi turistautak felmérését (*1. ábra*), bejárva ennek során a turistautak többségét [Kiss et al., 2007]. A felmérés első szakaszában, 2010 és 2011 között a magyarországi, 2014-ben a szlovákiai oldalon végeztem mérést.

A felméréshez az ELTE Térképtudományi és Geoinformatikai Tanszékéről kölcsönzött kézi GPS-t használtam. A könnyebb tájékozódás érdekében a turistautak egy-egy helyszínén fotót készítettem. A GPS-es felmérés megbízható alapként szolgált a turistautak felvételéhez. A fotók és a mérési jegyzőkönyv segítségével elkészítettem a turistautak nyomvonalát Google Earth-ben.



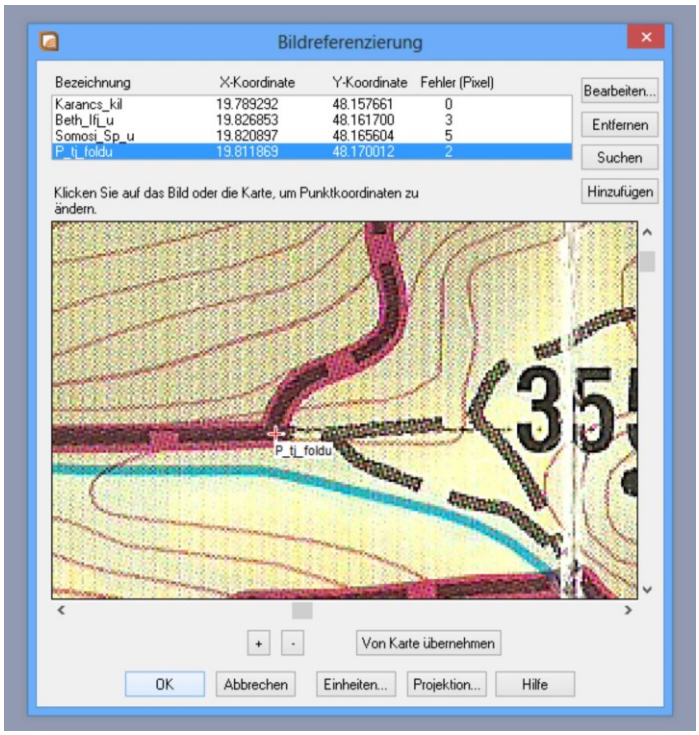
1. ábra A mérési terület vázlatos lehatárolása

2.2 Adatfelvétel térképről

A turistautak terepi bejárása jelentős energiát vett igénybe a dolgozat elkészítésénél. Emiatt a felmérést egy-egy nehezen járható vagy mérés által le nem fedett helyen térképi alapú adatfelvétellel (digitalizálás) egészítettem ki. Ehhez georeferálnom kellett a forrástérképeket [Kovács et al., 2002], [Cartographia, 2010].

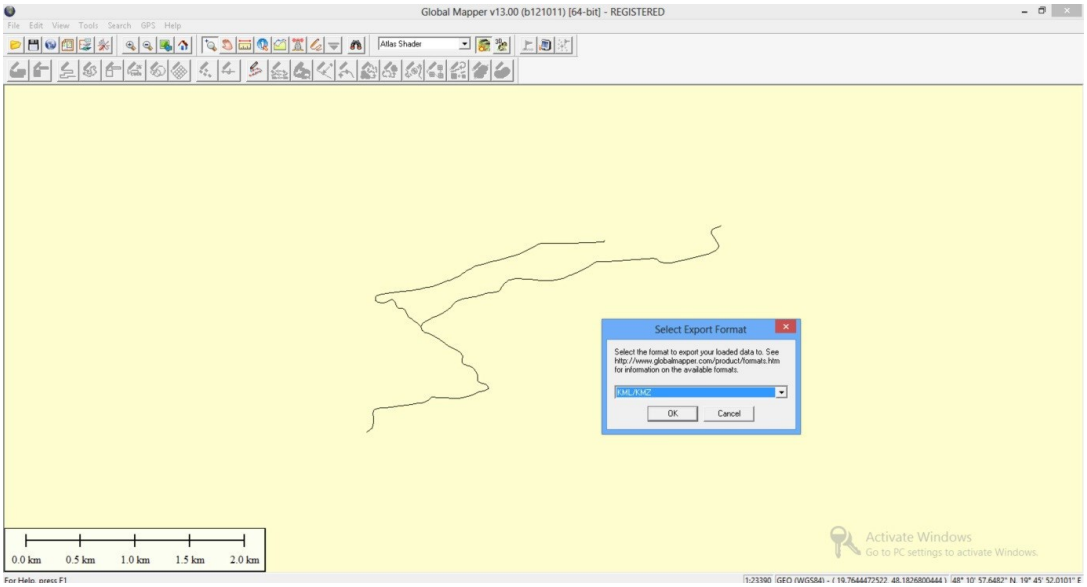
A georeferálás célja, hogy a térképet egy koordináta-rendszerbe helyezzük. Ehhez ismernünk kell adott számú pontjának koordinátáit a kívánt koordináta-rendszerben. A szükséges pontok száma transzformáció függő. Például Helmert-transzformáció esetén elegendő 2 pont ismerete, míg másodfokú polinomos illesztésnél 6 pont koordinátáit kell ismernünk. Az illesztőpontok mellett szükségünk van egy térformatikai szoftverre, ami az átszámítást elvégzi.

Munkám során két térképet georeferáltam. Mindkét esetben négy pont helyét jelöltem meg a térképen. A pontok koordinátáit a Google Earth szoftver segítségével nyertem ki. A transzformációt mindkét esetben földrajzi koordinátákkal végeztem, a MapInfo geoinformatikai program segítségével (*2. ábra*).



2. ábra Georeferálás a MapInfo felületén

A térképen szereplő turistautakat ezután átrajzoltam a MapInfo grafikai felületén. Az elkészült vonalat *.tab* formátumban tároltam. A célformátum viszont *.kml* volt, amellyel a Google Earth és a Google Maps is bánni tudnak. Szükségem volt egy programra, amely képes az egyik kiterjesztésből a másikba alakítani az adatokat. Ehhez a Global Mapper térinformatikai programot hívtam segítségül (*3. ábra*).



3. ábra .tab fájl exportálása .kml formátumba

3. Adatfeldolgozás

A GPS-sel nyert adatokat a GPS TrackMaker nevű szoftveren keresztül töltöttem le. Utóbb a program 13-as verzióját használtam. A GPS trackjét *.kml* formátumban mentettem. A *.kml* fájlt már meg tudtam nyitni Google Earth-ben.

A Google Earth műholdfelvételei, a GPS tracklogja és a digitalizált adatok alapján megrajzoltam a turistautak nyomvonalát Google Earth-ben. A kész térképi adatokat *.kml* fájlban tároltam. Így sikerült elérnem, hogy a két különböző forrásból származó adat egy formátumban szerepeljen.

3.1 A turista-útikalauz (2011) átalakítása

A 2011-ben BSc szakdolgozatként elkészített turista-útikalauz fő részei, így a begyűjtött útvonaladatok és a látnivalók adatai szolgálnak alapadatként a webes útikalauzhoz. Az új megjelenítési felület az OpenLayers keretrendszert használja választható Google vagy OpenStreetMap háttértérképpel. Az OpenLayers alkalmazása kisebb módosítást tett szükségessé a meglévő adatok szerkezetében.

Turistautak esetében korábbi munkámban rövid leírást készítettem Google Earth-ben az adott szakaszról. A leírás tartalmazta a szakasz nevét, rövid bemutatását, hosszúságát és végpontjainak szintkülönbségét.

Egy objektum leírása Google Earth-ben szöveg vagy HTML kód (Hypertext Markup Language) formájú lehet [Wikipédia, 2014]. A turista-útikalauzban a helyjelzők és útvonalak leírását HTML kóddal adtam meg, illetve CSS-sel (Cascading Stylesheets) formáztam [Wikipédia, 2014].

Mostani munkámban a turistautakat rövidebb részekre bontottam, ami könnyíti az útvonaltervezést. További eltérés, hogy kimaradnak a turistautakhoz tartozó szöveges leírások, helyüket az adott szakasz hosszúsága és egy hossz-szelvény veszi át. Az adatok ugyanúgy a megfelelő szakaszra kattintva hívhatók elő, mint Google Earth-ben.

A helyjelzőknél a látnivaló neve és általában leírás szerepelt a turista-útikalauzban. A leíráshoz helyenként kép is tartozott. A helyjelzőket mostani munkámba is átvettem, a szükséges változtatásokkal. A leírásokat és hiperlinkeket aktualizáltam, ahol nem találtam szükségesnek a képet, ott elhagytam, illetve mindenütt lecsökkentettem a képek méretét. Az ábrák átméretezéséhez és tömörítéséhez a Microsoft Office Picture Managert használtam. A szövegdoboz méretét egységes pixelméretre állítottam. A szövegdoboz tartalmára vonatkozóan ez 240 pixeles szélességet és 100 pixeles magasságot jelent.

A turistautak jelét az adott jelzés színével egyező színre állítottam. Az éppen kiválasztott útvonal vastagabb, 5 pixel szélességű vonallal jelenik meg (*4. ábra*). Új helyjelző vagy útvonal kiválasztásakor a kijelölést megszüntetjük, és visszaállítjuk a vonal alapértelmezett stílusát. Alapértelmezésben a vonalak szélessége 2 pixel. A kattintás pontosságának toleranciája 5 pixel. A szövegdoboz szélessége turistautak esetében is 240 pixel. Ez a szövegdobozban lévő táblázat szélességével egyezik.

4. ábra Turistaút kiválasztása a webes útikalauzban

A szövegdobozok az egyes térképi elemekre kattintva jelennek meg. Turistautak esetében tartalmazzák a szakasz hosszúságát, a jelzés teljes hosszúságát, összes szintkülönbségét, szöveges leírást a jelzésről és az adott szakasz domborzati profilját. A hossz-számítást JavaScript-kód végzi, a térképen megjelenített *.kml* fájl pontjainak koordinátája alapján.

A turista-útikalauz elkészítésénél a Google Maps térképeit. A Google Maps térképei Mercator-vetületet alkalmaznak, emiatt az átszámítás mercator-vetületbeli koordinátákkal történik [Gede, 2014].

A hossz-számítás mellett, szintén JavaScript-kód és a Google Elevation API segítségével elkészítünk egy kereszt-szelvényt a turistaút szintkülönbségeinek kirajzolására. A szelvény a kijelölt turistaút hosszában, a vonal pontjainak koordinátája és a Google domborzatmodellje alapján áll elő.

A Google többféle domborzatmodellt használ. Ezek egyike az SRTM (Shuttle Radar Topography Mission) domborzatmodell, amely a szárazföldi területek északi 60. és déli 57. szélessége közötti területre tartalmaz magassági adatokat [Molnár, 2008]. Felépítése szerint DEM típusú modell, vagyis az adatokat a Föld felszínére homogén módon tartalmazza [Elek, 2006]. Pontossága változó, hegységek területén nagyobb. Ahol elérhető nagyobb felbontású magassági modell, ott a Google azt használja. Ilyen lehet például a LIDAR (lézerradar) [Wikipédia, 2013] mérésekből származó modell. A Karancs területére 5 ismert magasságú pont összehasonlításával, +2–-9 méteres pontosságot kaptam.

Munkámban a diagram elkészítéséhez a Google Maps Javascript API (Application Programming Interface) ’visualization’ nevű alkönyvtárát használtam [Wikipédia, 2014]. Ennek segítségével adatainkból ábrákat készíthetünk.

4. Közzététel – turista-útikalauz a weben

4.1 Weboldal

A honlapot a W3C új szabványa, a HTML5 szerint kezdtem el készíteni [W3C Working Group Note, 2013]. A nyitóoldal létrehozásánál az egyszerűségre törekedtem. A két menüpont az útvonaltervezést és az adatletöltést valósítja meg.

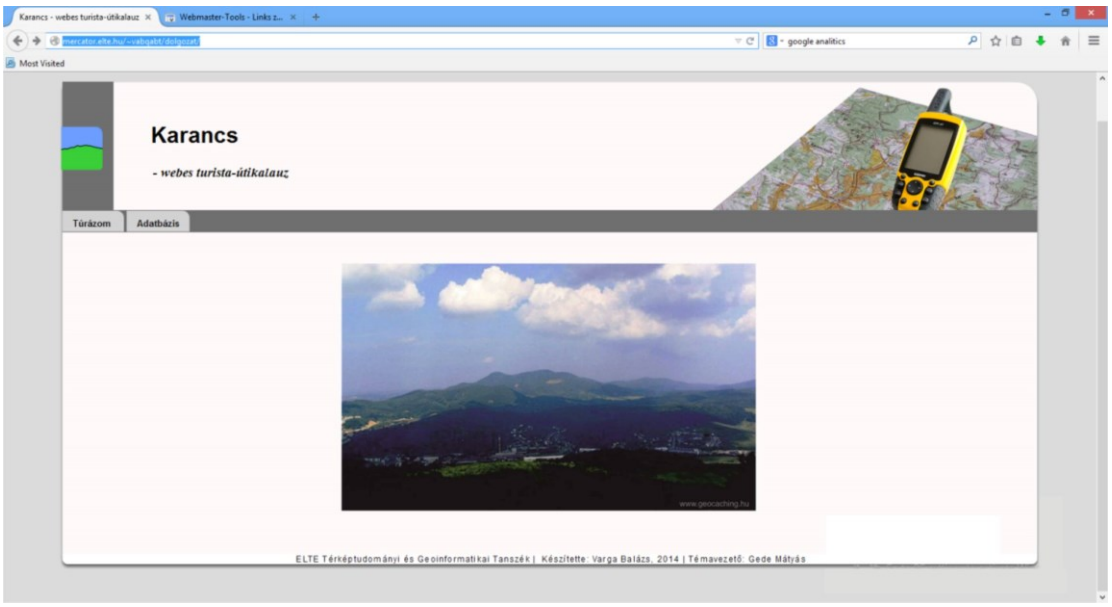
Az első menüpont alatt az OpenLayers beépülő modulján a turistautakat és látnivalókat tartalmazó térképi felületet érjük el. A turistautak kiválasztásánál a hosszúsági adat, a szintkülönbség és a magassági grafikon alapján megbecsülhetjük az útvonal nehézségi fokát. A turistajelzésekhez tartozó leírás pedig kiegészítő információval szolgál a felhasználó számára (jelzés kiindulási és végpontja, terepviszonyok). A helyjelzők az út mentén található látnivalókról tájékoztatnak, és a turista számára fontos információt tartalmazzák. Négyféle háttértérkép közül választhatunk. Ebből három a Google és egy az OpenStreetMap térképi szolgáltatása. Ezek segítségével felmérhetjük a közlekedés, a domborzat és a növényzet viszonyait. A könnyebb kezelhetőség érdekében a turistautak és fontos pontok külön ki-be kapcsolhatók a jobb oldalon található rétegkezelő menüvel.

A második menüponton keresztül a *.kml* és *.gpx* fájlok adatbázisához jutunk. A térképen szereplő útvonalak és fontos pontok innen letölthetők a Google Earth fájlformátumában (*.kml*) és a GPS-ek által támogatott formátumban (*.gpx*) [Wikipédia, 2013]. Az útszakaszok egyben és külön-külön is letölthetők. A fontos pontokat egy fájlban helyeztem el.

Az elkészült *.html* fájlokat és a stílusleíró *.css* fájlt az ELTE Térképtudományi és Geoinformatikai Tanszék szerverén lévő tárhelyemre töltöttem fel (*5. ábra*). Az oldal URL-je (Universal Resource Locator) *http://mercator.elte.hu/~vabqabt/dolgozat* lett.

A keresőoptimalizálásnál a ’Karancs, turista, útikalauz, térkép, turistaút, KML, GPX, letöltés, ELTE’ kulcsszavakat adtam meg. A keresőoptimalizálás célja, hogy a keresőmotorok minél kedvezőbb helyre sorolják be oldalunkat. Vagyis a megfelelő kulcsszavak megadásával elérhető legyen weboldalunk a Google, Bing, stb. keresőkből. További cél, hogy a találati listán minél jobb helyen szerepeljen a weboldal. A megfelelő kulcsszavakon kívül számít az oldalunk címe, amely az oldal fejlécében, a ’<title></title>’ elemek között áll. Az oldal leírásánál megadott szöveg - ’<meta name="description" content="…">’ - a találatkor felsorolásra kerülő snippetben, az oldal leírását tartalmazó címkében jelenik meg (snippet: darabka, töredék angol szóból) [Mueller, 2007]. A keresőoptimalizálás sikerességét ellenőrizhetjük a Google Webmaster-Tools segítségével. Itt nyomon követhetjük az oldalunk

indexeltségi állapotát, az oldalunkon szereplő kulcsszavak súlyát, az oldalunkra mutató weboldalakat, stb.. Az indexeltségi állapot mutatja, hogy a weboldalunk hány különböző URL-je szerepel a Google weboldalakat tartalmazó listájában (index) [Martienssen, (2009)]. Ha rendelkezünk Google fiókkal, egy *.html* kiterjesztésű fájl le-, majd oldalunkra való feltöltésével elhelyezhetjük oldalunkat a Google Webmaster-Tools adatbázisába. Én is éltem ezzel a lehetőséggel.



5. ábra A weboldal nyitólapja

4.2 Hordozható útikalauz

Az oldal az egyszerű túrázót kívánja szolgálni. Megvalósítja az útvonaltervezés, a távolság- és szintkülönbségbecslés feladatát. Emellett kiegészítő információt nyújt a túrázónak a látnivalókról. és a túrázáshoz szükséges adatokkal szolgál: jelöli a vízvételi lehetőségeket, pihenőhelyeket, esőházakat, stb.. A felhasználó az útvonalak és fontos pontok adatbázisából letöltheti és magával viheti (például GPS-re mentve) az útikalauzt.

4.3 A Google, a Google Maps és az OpenLayers függvénykönyvtárak

Az útikalauzt eredetileg a Google Maps API-val akartam megvalósítani, de a térkép interaktivitását ez nem tette lehetővé, így az OpenLayers mellett döntöttem.

A Google Maps és az OpenLayers moduljai alapvető fontosságúak a turista-útikalauz technikai megvalósításánál [OpenLayers API dokumentáció, 2014]. Ez a két függvénykönyvtár tartalmazza a megjelenítést végző Javascript függvényeket, és ezek felhasználásával tudjuk felruházni térképünket interaktív funkciókkal. A Google Javascript API a domborzati profil megjelenítését biztosítja [Google Javascript API].

Az OpenLayers egy nyílt forráskódú, felhasználó oldali Javascript függvénykönyvtár, a MetaCarta nevű cég fejlesztése [Hazzard, 2011]. A modult jelenleg az OS Geo (Open Source Geospatial Foundation) felügyeli [Gede, 2014]. Az OpenLayers jelentősége, hogy különböző forrásokból származó térképet tudunk vele megjeleníteni egy egységes keretben. Az adatszolgáltató lehet WMS (Web Map Service) és WFS (Web Feature Service) szerver vagy a Google Maps, illetve még jó néhány más térképkiszolgáló. Állhat a háttérben adatforrásként *.kml* vagy *.geojson* fájl is. Ezek felhasználásával valós idejű interakciót tudunk megvalósítani adatainkon [Hazzard, 2011]. Tehát az OpenLayers egy programozói felület, amin térképes alkalmazásokat készíthetünk.

A turista-útikalauzban a térképi adatokat a Google és az OpenStreetMap szolgáltatja. A Google térképei tartalmazzák a műholdképek mozaikját és másik két térképet, amelyek közül az egyik egy Google-féle autótérkép, a másik egy summerolt domborzatot és az utakat ábrázoló térkép.

4.4 A megjelenítés eszköze, az OpenLayers API

Kódunk működésének hátterében a honlap fejlécében hivatkozott három JavaScript fájl áll. Ezek közül első az OpenLayers könyvtára [Hazzard, (2011)].

```
<script src="http://openlayers.org/api/OpenLayers.js"></script>
<script src="http://www.google.com/jsapi"></script>
<script src="http://maps.google.com/maps/api/js?v=3.2&sensor=false&libraries=geometry">
</script>
```

Az OpenLayers névtér tartalmazza a megjelenítéshez szükséges osztályokat és azok objektumait. Ezeket felhasználva objektum orientált szemlélet szerint programozhatunk: definiálhatjuk az objektumok tulajdonságait, és megváltoztathatjuk őket egy-egy metóduson keresztül [Eric Hazzard (2011)]. Az OpenLayers objektummodelljét az OpenLayers felhasználói oldala tartalmazza [OpenLayers API dokumentáció, 2014].

A kód második sora a Google JavaScript API-ra hivatkozik. A Google számos függvénykönyvtárát ezen keresztül érhetjük el.

A Google Maps API-val a Google térképeit építhetjük be weboldalainkba. A Google Mapsre könyvtárra való hivatkozás a harmadik sorban szerepel.

A turista-útikalauz fő elemei a térkép (’Map’), a térképi réteg (’Layer’) és a vetület (’Projection’) osztályok, és ezek objektumai. Alább az oldal betöltődésekor végrehajtódó függvény egy részletét látjuk, amellyel létrehozzuk a térkép osztály egy példányát a ’map’ (globális) változó alatt, és egy vektoros adatok tárolására alkalmas térképi réteget. A vektor réteg konstruktor függvényében megadhatjuk a réteg nevét és beállításait. A lenti példában ilyenek az ’isBaseLayer’, ’strategies’ és ’protocol’ tulajdonságok. Ezekkel beállíthatjuk, hogy egy réteg alaprétegként vagy fedvényként szerepeljen a térképünkön. Alaprétegek esetén az ’isBaseLayer’ tulajdonságot ’true’-ra kell állítanunk. Ez azt jelenti, hogy a térképünkön folyamatosan be van kapcsolva. Fedvények (overlay) esetében az ’isBaseLayer’ tulajdonság ’false’ értékű kell legyen és a térkép ’transparent’ tulajdonságát pedig ’true’-ra kell állítanunk. Mindkét típusú rétegből (baselayer, overlay) többet is definiálhatunk. Beállíthatjuk, hogy rétegünk képes legyen-e betöltődés után újabb adatelérésre, illetve megadhatjuk a protokollt és azon belül a megjeleníteni kívánt fájl URL-jét és formátumát.

```
…
function init() {
        map = new OpenLayers.Map('map',{});
        track = new OpenLayers.Layer.Vector("Túraútvonal",
                {
                isBaseLayer: false,
                strategies: [new OpenLayers.Strategy.Fixed()],
                protocol: new OpenLayers.Protocol.HTTP(
                        {
                        url: "data/karancs_tj_osszes_20140411.kml",
                        format: new OpenLayers.Format.KML({ extractStyles: true })
                        })
                });
                …
        }
…
```

A rétegek hozzáadása térképünkhöz csak ezután következhet (ld. lentebb). A térképhez először a Google és az OpenStreetMap térképi rétegeit adjuk hozzá, majd az általunk létrehozott adatokat használó térképi rétegeket (’track’ és ’poi’). Láthatjuk, hogy az OpenLayers egy külön rétegtípust biztosít a Google térképmozaikok megjelenítésére, amely a Google Maps API függvényeit használja. Ezért kell annak scriptjei is csatolni a weboldalhoz. Ennek köszönhetően a Google Maps ″becsomagolható″ az OpenLayers megjelenítő felületbe.

```
function init() {
        …
        map.addLayer(new OpenLayers.Layer.Google("Google Hybrid",{type:
google.maps.MapTypeId.HYBRID, numZoomLevels: 22}));
        map.addLayer(new OpenLayers.Layer.Google("Google Road Map",{type:
google.maps.MapTypeId.ROADMAP, numZoomLevels: 22}));
        map.addLayer(new OpenLayers.Layer.Google("Google Terrain",{type:
google.maps.MapTypeId.TERRAIN, numZoomLevels: 22}));
        map.addLayer(new OpenLayers.Layer.OSM());
        map.addLayer(track);
        map.addLayer(poi);
        ...
```

A ‘LayerSwitcher’ kezelőszerv segítségével a baselayertől különböző típusú rétegeket kapcsolhatjuk térképünkön egy oldalsó panel segítségével. A kezelőszervet az ‘addControl()’ metódussal adhatjuk térképünkhöz. A ‘zoomToExtent()’ metódus a paraméterként megadott téglalap értékeihez állítja be a térképablakot.

```
        …
        map.addControl(new OpenLayers.Control.LayerSwitcher());
        map.zoomToExtent(new OpenLayers.Bounds(2200000,6125000,2205000,6140000));
        …
```

A térkép interaktivitását megvalósító funkciók ezután következhetnek. Itt két függvény kap hangsúlyt. Az elsővel egy információs ablakot jelenítünk meg az elemekre való kattintáskor (‘function showDescription()’), a másik a keresztszelvény kirajzolását végzi (‘function plotElevation()’).

Az OpenLayers ‘SelectFeature()’ kontrolljával eseménykezelőket rendelünk a két megadott vektoros réteg elemeinek kijelöléséhez. A vektoros elemek kiválasztásánál (‘onSelect’) egy olyan függvényt hívunk meg, amely kiírja a jelzés vagy a fontos pont leírását.

```
…
var sfc=new OpenLayers.Control.SelectFeature([track,poi],
{
        toggle: true,
        onSelect: function(feature)
        {
        showDescription(feature,this.handlers.feature.evt)
        },
…
```

Helyjelzők esetében ezek a név és a leírás. Útvonalak esetén a név, a hosszúsági adatok, a leírás és egy keresztszelvény. A leírás (‘<description>’) tag nem mindig tartalmaz adatot a *.kml* fájlban, így csak akkor akarjuk kiíratni, ha nem üres. A JavaScript-kódban ezt egy feltételes utasítás beiktatásával oldottam meg:

```
…
var isDesc = feature.attributes.description ? feature.attributes.description : "";
tartalom=+'<h3>'+nev+'</h3>'+'<span>Szakasz hossza: '+kbKmHossz+' km</span>'+isDesc+'<div
id="elevation_chart" style="height:120px;width:240px;margin:0 -2em 0 0;padding:0;"
alt="elevation_chart"></div>';
…
```

Az utasítás első paramétere egy boolean típusú (igaz vagy hamis) kifejezés. Ha igaz a kifejezés a kérdőjel utáni értéket kapjuk eredményül, ha nem, akkor pedig a kettőspont utánit. A kifejezés a második esetben jelenleg egy üres stringet (karakter típusú változót) ad vissza.

A hosszúság kiszámítása a *.kml* fájlban hozzáadott töröttvonal pontjainak koordinátái alapján lehetséges. A programkód az egyes szakaszok hosszának kiszámításához minden két töréspontra Pitagorasz-tételt alkalmaz, és ezeket egy ciklus segítségével összegzi. Az eredményt kerekítést követően írja ki.

A hosszúságszámítást végző képlet a teljes vonalszakaszra:

$$\sum_{i=1}^{l} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$$, ahol

(x_i, y_i) - a töröttvonal *i*-edik töréspontjának vetületi koordinátái

l - a töréspontok száma

Ezután a pontok koordinátáit transzformáljuk a térkép által alkalmazott Mercator-vetület koordinátáiból földrajzi koordinátákká. Ezt azért tesszük, mert a magassági profil kirajzolásához földrajzi koordinátákra van szükségünk. A koordinátákat páronként eltároljuk a 'nyomvonal[]' nevű tömbben.

A hosszúságszámítást, vetületi transzformációt és a földrajzi koordináták eltárolását végző kódrészlet:

```
for(var i=1;i<feature.geometry.components.length;i++)
{
        hossz+=Math.sqrt(Math.pow(feature.geometry.components[i].x-
        feature.geometry.components[i-1].x,2)+Math.pow(feature.geometry.components[i].y-
        feature.geometry.components[i-1].y,2));

                foldrajzi=feature.geometry.components[i].clone().transform(merc,wgs84);
                nyomvonal.push(new google.maps.LatLng(foldrajzi.y,foldrajzi.x));

}
```

A hosszúságot sok tizedesjegy pontossággal kapjuk, így célszerű kerekítenünk. A megkapott hosszat két tizedesjegyre kerekítettem a 'Math.round()' függvény felhasználásával, az alábbi módon:

```
kbKmHossz = Math.round(hossz/10)/100;
```

A túraútvonalhoz tartozó szövegdobozhoz most már hozzáadhatjuk a számított hosszúságértéket és a keresztszelvényt későbbiekben tartalmazó ' ' elemet.

Még ezen a kódrészleten belül meghívjuk a Google Elevation Service 'getElevationAlongPath()' nevű függvényét, amellyel egy útvonal ('path:nyomvonal') adott számú pontjára kérdezhetjük le a magasságot. Az útvonal pontjait esetünkben a nyomvonal[] nevű tömb tárolja. A mintavételezés során a vonalat adott számú - most 512 db - egyenlő részre osztjuk és kapjuk az egyes osztópontokhoz tartozó magasságokat. (A magassági értékek lekérdezése tehát nem a töréspontok alapján történik.)

A függvény másik változója, a 'plotElevation()' metódus, amely a magassági adatok feldolgozását végzi. A magassági adatok lekérése aszinkron folyamat.

A ‘getElevationAlongPath()’ függvény egy http-kérést küld a Google-nek, de nem vár a válaszra; ha megérkezett a válasz a lekérésre, akkor meghívódik egy callback függvény az adatok visszaadására. Ez most a ‘plotElevation‘ metódus. Ennek definiálása később következik.

A szövegdoboz tartalmát kiíró kódrészlet és a magassági adatok lekérdezése, a callback metódus hívásával:

```
        …
        tartalom='<h3>'+nev+'</h3><span>Szakasz hossza: '+kbKmHossz+' km</span>'+isDesc+'<div
        id="elevation_chart" style="height:120px;width:240px;margin:0 -2em 0 0;padding:0;"
        alt="elevation_chart"></div>';
        elSvc.getElevationAlongPath({path:nyomvonal,samples:512}, plotElevation);
    }
    …
```

Az ‘showDescription()’ függvény végén létrehozzuk a szövegdobozt és megadjuk tulajdonságait. A szövegdobozt az OpenLayers különféle ‘popup’ osztályai közül a ‘FrameCloud’-dal valósítjuk meg.

A ‘Size’ objektumban megadjuk a doboz szélességét és magasságát pixelben, majd az objektumot tartalmazó változót elhelyezzük a ‘FramedCloud’ harmadik paraméterébe. A ‘tartalom’ változóban beillesztésre kerül a korábban definiált tartalom (név, hossz, leírás, szelvény), illetve az ‘ll’ változó alatt a kattintás koordiánái. Utóbbi a térképen a szövegdoboz vonatkozási helyét jelöli. Végül az ‘addPopup()’ metódussal hozzáadjuk a térképünkhöz az információs ablakot.

```
    …
    var meret=new OpenLayers.Size(280,300);
    var p=new OpenLayers.Popup.FramedCloud('infoPopup',ll,meret,tartalom,null,true);
    map.addPopup(p);
    }
    ...
```

4.5 Diagram készítése, a Google Visualization API

Programunk másik fő feladatát a ’plotElevation()’ függvény végzi, amely felhasználva a Google Elevation Service által szolgáltatott adatokat és a Google Visualization könyvtárat kirajzolja a már említett keresztszelvényt (3.1 rész) [Google Visualization API, 2014]. A ElevationService szolgáltatása a Google Maps API harmadik verziójában közvetlenül az ElevationService() objektumból is elérhető.

A függvény az Elevation Service által küldött válasz megérkezésekor hívódik meg. Első argumentumában (’results’) kapja a lekérdezett magassági adatokat, míg a második (’status’) változó a lekérdezés sikerességéről ad visszajelzést. Egy elágazás beiktatásával megadjuk, hogy a magassági adatok sikeres elérése esetén azok az ’elevations’ nevű tömbben tárolódjanak. Következhet a diagram adattáblájának elkészítése.

```
function plotElevation(results, status) {
        if (status == googleMaps.ElevationStatus.OK) {
          elevations = results;

          var data = new google.visualization.DataTable();
          data.addColumn('string', 'Sample');
          data.addColumn('number', 'Tengerszint feletti magasság (m)');
                for (var i = 0; i < results.length; i++) {
                data.addRow(['', elevations[i].elevation]);
                }
          …
      }
}
```

Az API fontos eleme a ’DataTable()’ nevű osztály, amely egy adattáblát reprezentál. A kódban (ld. alább) először létrehozzuk az adattábla egy példányát. Ezután definiáljuk a diagram oszlopait az ’addColumn()’ függvénnyel. A függvény első változója az adattípust, második változója az oszlop nevét adja meg. Végül a magassági lekérdezés adatait egy ciklus felhasználásával soronként az adattábla nevű objektumba töltjük. A diagram létrehozása és megrajzolása csak ezután következhet.

```
…
var data = new google.visualization.DataTable();
data.addColumn('string', 'Sample');
data.addColumn('number', 'Tengerszint feletti magasság (m)');
          for (var i = 0; i < results.length; i++) {
          data.addRow(['', elevations[i].elevation]);
          }
…
```

A magassági profilt a JavaScript-kód az eltérő a magasságú vonalak egymás mellé illesztésével rajzolja ki (ld. alább). Az oszlopdiagram egy példányát előállítjuk és elhelyezzük a szövegdoboz 'elevation_chart' azonosítójú ’ ’ elemében. A keresztszelvényt a diagram

’draw()’ nevű metódusával rajzoltatjuk ki az adattábla adatai alapján, a megadott tulajdonságok, esetünkben szélesség (’width’) és magasság (’height’) szerint. A ’legend’ tulajdonság segítségével beállítjuk, hogy ne jelenjen meg jelmagyarázat. A diagram méretére vonatkozó adatok pixelértékben értendők.

```
chart = new google.visualization.ColumnChart(document.getElementById('elevation_chart'));
chart.draw(data,
                {
                width: 240,
                height: 120,
                legend: 'none'
                });
```

5. A turista-útikalauz fájlformátumai

A következő részben azt a két fájlformátumot mutatom be, amely a turista-útikalauz honlapján a Letöltések rész alatt az összegyűjtött adatot tartalmazza. A *.kml* kiterjesztés a Google Earth fájlformátuma, míg a *.gpx* a GPS és a számítógépes alkalmazások közötti kommunikációt szolgálja.

5.1 A .kml kiterjeszés

A KML (Keyhole Markup Language) a Keyhole Inc. cég által létrehozott, XML-struktúra szerint felépülő adatformátum. Elemeinek segítségével földrajzi vonatkozású információt tárolhatunk. Például pontok, vonalak, sokszögek földrajzi adatait [Google KML dokumentáció, 2013]. KML fájlt legegyszerűbben a Google Earth grafikus funkcióival hozhatunk létre. Ha ismerjük a KML nyelvet, Jegyzettömbben is készíthetünk ilyen fájlt.

Helyjelző megadása:

```
<kml>
<Document>
        ...
        <Placemark>
                <name>Liszt Ferenc Nemzetközi Repülőtér</name>
                <description>Magyarország legnagyobb forgalmú repülőtere.</description>
                <LookAt>
                        <longitude>19.26405249177911</longitude>
                        <latitude>47.41334774757686</latitude>
                        <altitude>0</altitude>
                        <heading>-0.5546734342823477</heading>
                        <tilt>0.0327193523377881</tilt>
                        <range>31169.79668817877</range>
                        <gx:altitudeMode>relativeToSeaFloor</gx:altitudeMode>
                </LookAt>
                <styleUrl>#m_ylw-pushpin</styleUrl>
                <Point>
                        <gx:drawOrder>1</gx:drawOrder>
                        <coordinates>19.25467158827829,47.43095823624149,0</coordinates>
                </Point>
        </Placemark>
</Document>
</kml>
```

A helyjelző adatait a ’<Placemark>’ tag tartalmazza. Ezen belül megadhatjuk a megjelölt pont nevét (’<name>’), leírását (’<description>’) és beállíthatjuk a megjelenítés paramétereit (’<LookAt>’). A megjelenítés paramétereihez tartoznak, hogy a világ melyik pontjára nézünk (’<longitude>’,’<latitude>’), a Föld felszínéhez képest viszonyított magasság ’<altitude>’ és a rálátás beállításai. A rálátás paramétereiből az első kettő a rálátás szögei, a ’<range>’ pedig a kamera magasságát adja meg méterben a definiált ponttól számítva. Végül

az ’<altitude>’ tag vonatkozási helyét (’<gx:altitudeMode>’) adhatjuk meg. A vonatkozási viszony háromféle lehet: clampToGround (közvetlenül a föld felszínén lévő), relativeToGround (az ’<altitude>’ tagben megadott magasságban a föld felett), illetve absolute (az ’<altitude>’ tagben megadott magasságban a tengerszint felett).

A ’<Point>’ paraméter alatt az elem helyét definiálhatjuk a grafikai sorrendben (’<gx:drawOrder>’), valamint a pont koordinátáit és magasságát (’<coordinates>’). A grafikai sorrendnél a magasabb értékű pont kerül feljebb [Google KML dokumentáció, 2013].

Fájl felépítése LineString típusú objektum megadásakor:

```
…
<Placemark>
        <name>Északi zöld 1</name>
        <description>Északi zöld jelzés: Somoskőújfalu vá.–Karancs kilátó.</description>
        …
        <styleUrl>#inline01403</styleUrl>
        <LineString>
                <extrude>0</extrude>
                <tessellate>1</tessellate>
                <altitudeMode>absolute</altitudeMode>
                <coordinates>
                        19.81603504579919,48.1549214156935,0
                        19.81528472531372,48.15503299250817,0
                        19.81506818075378,48.15511435683085,0
                        19.81521763118587,48.15529902360499,0
                        19.8154731634552,48.1557580252721,0
                        ...
                </coordinates>
        </LineString>
</Placemark>
</Document>
</kml>
```

A különbség a pont megadásához képest, hogy ’<LineString>’-nél megadhatjuk a Föld felszínhez való illeszkedést, a ’<tessellate>’ taggel. Ennek értéke alapértelmezésben 0. Ha el szeretnénk érni a kívánt a funkciót, értékét 1-re kell definiálnunk. Az ’<extrude>’ elem akkor kap hangsúlyt, ha a vonal tengerszint feletti magassága nagyobb, mint 0. Ekkor a vonalat töréspontjai mentén a Föld felszínére vetíthetjük. Ehhez az alapértelmezett érték, 0 helyett 1-et kell megadnunk.

Helyjelző, Útvonal és Poligon típusú objektum, illetve Lefedő kép esetén a *.kml* dokumentumban megadhatunk stílusdefiníciót. A fenti kódrészletben a ’<styleUrl>’ kódelemben stílusra vonatkozó hivatkozást látunk. A stílust értelemszerűen a stílushivatkozás előtt kell megadnunk. A *.kml* dokumentumban definiált stílusokra egy # jellel lehet hivatkozni. Ehhez a ’<Style>’ tagben az ’id’ paraméter után egy azonosítót kell neki

definiálunk. Egy stílust különböző típusú elemeknél is felhasználhatunk. Ezek az ún. közös használatú stílusok (*shared styles*) [Google KML dokumentáció, 2013].

Stílusdefiníció helyjelzőre:

```
<Style id="highlightPlacemark">
    <IconStyle>
     <Icon>
       <href>http://maps.google.com/mapfiles/kml/paddle/red-stars.png</href>
     </Icon>
    </IconStyle>
   </Style>
   <Style id="normalPlacemark">
    <IconStyle>
     <Icon>
       <href>http://maps.google.com/mapfiles/kml/paddle/wht-blank.png</href>
     </Icon>
    </IconStyle>
   </Style>
   <StyleMap id="exampleStyleMap">
    <Pair>
     <key>normal</key>
     <styleUrl>#normalPlacemark</styleUrl>
    </Pair>
    <Pair>
     <key>highlight</key>
     <styleUrl>#highlightPlacemark</styleUrl>
    </Pair>
   </StyleMap>
```

A fenti példában látható, hogy egy helyjelzőre párhuzamosan két kinézetet is megadhatunk. Ezeket a ‘<StyleMap>’ elemben a ‘<key>’ elem két különböző értékével – ‘normal’ vagy ‘highlight’ – különíthetjük el. A ‘normal’ stílus az alapértelmezett, a ‘highlight’ kulcs alatti pedig a Helyjelző fölé görgetéskor jelenik meg.

Stílusdefiníció Útvonalra és Poligonra:

```
<Style id="utvonal">
  <LineStyle>
    <width>1.5</width>
  </LineStyle>
</Style>
<Style id="sokszog">
 <PolyStyle>
    <color>7dff0000</color>
  </PolyStyle>
</Style>
```

Az első stílusdefinícióban egy Útvonal szélességét adtuk meg pixelértékben. A második definíció egy színértéket jelöl egy Poligon esetén.

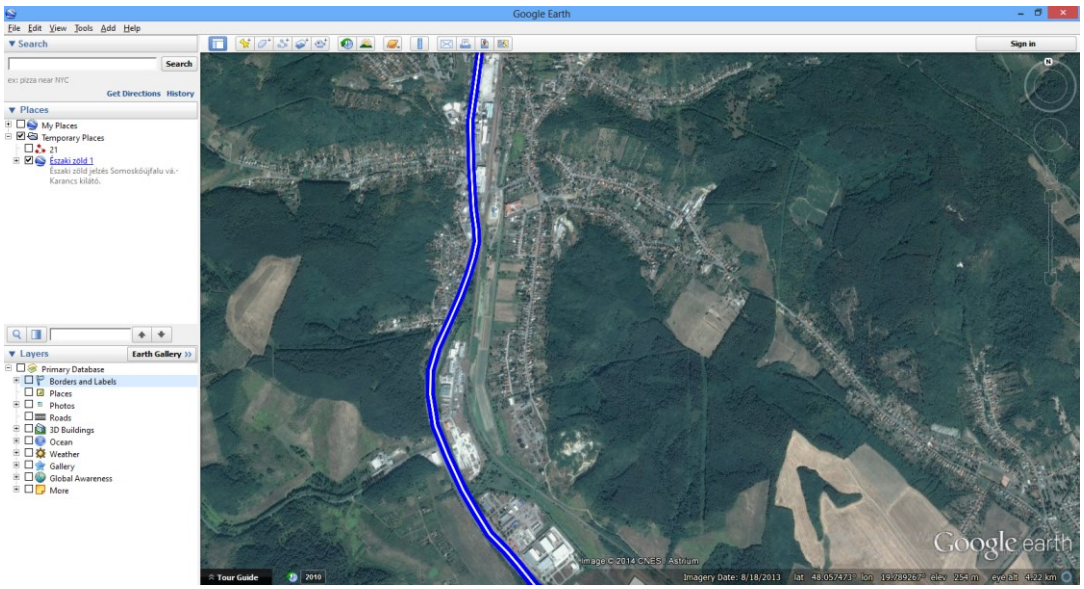
További tulajdonságok megadása lehetséges, így megadhatjuk a vonal szélességét méterben a <gx:physicalWidth> elemmel. A vonal feliratozása a ‘<gx:labelVisibility>’ tagen belül, 0 és 1 érték megadásával kapcsolható ki vagy be. A felirat a ‘<name>’ elemen belül

megadott megnevezést tünteti fel, vonalra vonatkoztatott névként. Készíthetünk kontúrral ellátott vonalat is (*7. ábra*). Erre mutat példát az alábbi kódrészlet:

```
<LineStyle>
        <color>ffffffff</color>
        <gx:physicalWidth>50</gx:physicalWidth>
        <gx:outerColor>ffff0000</gx:outerColor>
        <gx:outerWidth>0.8</gx:outerWidth>
</LineStyle>
```

Négy paramétert adtunk meg a vonalstílus négy sorában: közbenső vonal színe, (külső) vonal szélessége, (külső) vonal színe, közbenső vonal szélessége a külső vonal szélességéhez viszonyítva. A <gx:outerWidth> 0.8-es érték esetén a belső vonal szélességét a külső vonal szélességéhez képest 20%-osra állítja.

Nem esett még szó arról, hogy pontosan hogyan adhatunk meg színeket, illetve átlátszóságot. E két tulajdonságot egyszerre adhatjuk meg színek definiálásánál. A színeket egy 8 jegyű hexadecimális számsorral adhatjuk meg, ahol minden két jegy egy színt, illetve átlátszóságot kódol, a következő sorrendben: aa (átlátszóság), bb (kék szín), gg (zöld szín), rr (vörös szín). Ennek alapján kiszámolható, hogy színenként 16 x 16, összesen pedig $(16 \times 16)^4$, vagyis 2^{32}-féle színt adhatunk meg.



7. ábra Kontúrozott vonal Google Earthben a fenti példa alapján

5.2 A GPX fájlformátum

Az adatok GPS-be való feltöltéséhez a turistautakat és fontos pontokat a szintén XML alapú GPS Exchange (*.gpx*) formátumban is elhelyeztem az útikalauz honlapján [Wikipédia, 2013].

Részlet a turistautakat tartalmazó *.gpx* fájlból:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<gpx
 version="1.1"
 creator="Global Mapper - http://www.globalmapper.com"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://www.topografix.com/GPX/1/1"
 xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com/GPX/1/1/gpx.xsd">

 <metadata>
 </metadata>

<trk>
 <name><![CDATA[Piros háromszög 1]]></name>
 <desc>Unknown Line Type</desc>
 <trkseg>
  <trkpt lat="48.144103" lon="19.800977">
  </trkpt>
  <trkpt lat="48.143973" lon="19.801111">
  </trkpt>
  <trkpt lat="48.143729" lon="19.801172">

  …
<extensions>
 </extensions>
</gpx>
```

A fenti fájl struktúrája:

- dokumentum típus defininíció,
- leíróadatok (‘<metadata>’),
- útvonalak (‘<trk>’),
- az útvonalneve (‘<name>’), útvonal leírása (‘<desc>’), és az útvonal szegmensei (‘<trkseg>’),
- az egyes pontok (‘<trkpt>’), mint vonalszegmensek, földrajzi koordinátákkal megadva,
- a kiterjesztések (‘<extensions>’), az egyes gyártók által használt további paraméterek [Wikipédia, 2014]

  Például: cím, telefonszám vagy egyéb paraméterek (vízmélység, hőmérséklet stb.)

A minimálisan megadandó paraméterek GPX fájl esetén az egyes pontok koordinátái, minden további elem használata opcionális.

Összefoglalás

A Karancs webes turista-útikalauzának elkészítésénél egy túrázást segítő weboldal létrehozása volt a célom, amely segítségével felmérhetők a távolság és szintkülönbség viszonyai a túrázó számára, és amely segítséget nyújt azoknak, akik még nem jártak ezen a terepen.

A feladatot a Google Maps API-ban kezdtem el megvalósítani, viszont annak korlátoltsága miatt, így az interaktivitás hiánya miatt, az OpenLayers-t választottam megjelenítő felületként.

Az OpenLayers és a Google API-jainak felhasználásával sikerült megvalósítani egy nyílt elérésű térképeken alapuló és nyílt forráskódú technológiára támaszkodó egyszerű webes térképi alkalmazást. A további terepi felvételnek köszönhetően pedig a turistautak Magyarország és Szlovákia területére is elérhetők.

Hivatkozott irodalom

Könyv

Elek I., 2006. Bevezetés a geoinformatikába. *ELTE Eötvös Kiadó, Budapest*, p. 162, 172

Hazzard E., 2011. OpenLayers 2.10, Beginner's Guide. *Packt Publishing, Birmingham-Mumbai, p. 8, 24, 228*

Kiss G., Baráz Cs., Gaálová K., Judik B., 2007. A Karancs–Medves és a Cseres-hegység Tájvédelmi Körzet. *Bükki Nemzeti Park Igazgatóság, Eger, p. 10.*

Szakdolgozat

Varga B., 2011. A Karancs hegység háromdimenziós turista-útikalauza Google Földben, alapszakos szakdolgozat. *ELTE, Budapest*

Térkép

Cartographia, 2010. Cserhát térkép, 2010, 1 : 60 000, *Cartographia, Budapest*

Dornyai SE, 2002. Karancs tájfutó- és turistatérkép, 2002, 1 : 15 000, *Dornyai SE, Salgótarján*

Weboldal

Gede Mátyás: Webes geoinformatika 1. gyakorlat, elektronikus jegyzet, (2014)
http://mercator.elte.hu/~saman/hu/okt/ol/

Gede Mátyás: Script nyelvek alkalmazása a web-kartográfiában, elektronikus jegyzet, (2014)
http://mercator.elte.hu/~saman/hu/okt/gmaps3/

Molnár Gábor: Földkutatás a világűrből, elektronikus jegyzet, (2008)
http://sas2.elte.hu/mg/foldkutatas_v3/11radar3srtm.htm

Google Earth Súgó, (2014)
https://support.google.com/earth/answer

Google Javascript API, (2014)
https://code.google.com/p/google-api-javascript-client/

Google KML dokumentáció, (2013)
https://developers.google.com/kml/documentation

Google Maps API V3, dokumentáció (2014)
https://developers.google.com/maps/documentation/javascript/reference

Google Visualization API, (2014)
https://developers.google.com/chart/interactive/docs/reference

Google Webmaster-Tools felülete (Google fiókba való bejelentkezés szükséges)
https://www.google.com/webmasters/tools/

John Mueller: Webmaster-Zentrale, Meta-Tags und Web Search, szócikk, (2007)
http://googlewebmastercentral-de.blogspot.hu

Martienssen Alf: Google-Tools, online cikk, (2009)
http://seo-hungary.com/2009/08/google-tools/

OpenLayers API dokumentáció, (2014)
http://dev.openlayers.org/apidocs/files/OpenLayers-js.html

W3C Working Group Note, dokumentáció, (2013)
www.w3.org/TR/html-markup/

Wikipédia, GPX szócikk, (2013)
http://hu.wikipedia.org/wiki/GPX

Wikipédia, GPS Exchange Format szócikk, (2014)
http://en.wikipedia.org/wiki/GPS_Exchange_Format

Wikipédia, HTML szócikk, (2014)
http://hu.wikipedia.org/wiki/HTML

Wikipédia, CSS szócikk, (2014)
http://hu.wikipedia.org/wiki/CSS

Wikipédia, application programming interface szócikk, (2014)
http://en.wikipedia.org/wiki/Application_programming_interface

Wikipédia, lézerradar szócikk, (2013)
http://hu.wikipedia.org/wiki/L%C3%A9zerradar

A weboldalak utolsó elérésnek ideje: 2014. május

Nyilatkozat

Alulírott, **Varga Balázs (NEPTUN kód: WLMHE5)** nyilatkozom, hogy jelen szakdolgozatom teljes egészében saját, önálló szellemi termékem. A szakdolgozatot sem részben, sem egészében semmilyen más felsőfokú oktatási vagy egyéb intézménybe nem nyújtottam be. A szakdolgozatomban felhasznált, szerzői joggal védett anyagokra vonatkozó engedély a mellékletben megtalálható.

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

HOZZÁJÁRULOK NEM JÁRULOK HOZZÁ

Budapest, 2014. június 05.

……
a hallgató aláírása