# Pályaépítés nagyon kezdőknek OpenBVE-ben

Második, átdolgozott kiadás

(2.2-es verzió)

Legutóbbi frissítés: 2017. október 6.

# 1. rész

Az utóbbi időben elég sok megkeresést kapok, amelyben pályaépítésről érdeklődik a tisztelt levélíró. Így hát belefogtam, hogy egy használható leírást írjak. Ez itt az első része, teljesen az elejéről kezdjük. Néhány dolog szükséges lesz hozzá, ezek megléte alapkövetelmény:

- Számítógép Windows 10/8/7/Vista/XP, Linux vagy Mac operációs rendszerrel
- Jegyzettömb (Windowsban gyárilag telepítve) vagy Notepad++ (ingyenes és gyors szövegszerkesztő, letölthető innen: notepad-plus-plus.org). Linuxon megfelel a Gedit, a Kate, vagy más egyszerű szövegszerkesztő
- Némi józan paraszti ész, logikus gondolkodásmód, olvasni tudás, szövegértés, kitartás akár hónapokon keresztül
- OpenBVE program (minimum 1.4.3-as verzió), rajta működő állapotban a MÁV 120a (Budapest-Mende) pálya, valamint a V63-as (Gigant) villanymozdony, ugyanis innen fogjuk átvenni a hozzávalókat. Mindkettő letölthető a BVE Klub oldaláról

Ha ezek közül bármelyik is hiányzik, úgy nem megfelelő eredményt hozhat a próbálkozásunk, így igyekezzünk mindezek meglétére ügyelni! Mivel ez egy próbapálya, ezért kérek mindenkit, hogy az olvasott módon hajtsa végre az utasításokat. Igen, úgy, mint egy robot. A leírás célja az OpenBVE pályaépítési logikájára való rámutatás, működésének elmagyarázása. Ha végigértünk a leckékkel, akkor már lehet egyénieskedni, nekilátni a saját pályának, de addig nem javaslom.

Most, hogy tisztáztuk az alap dolgokat, fogjunk is bele az első leckébe, amelynek célja, hogy legyen egy pályánk, amin bármilyen vonatot tudunk futtatni. A pálya egyenesen fog haladni, mindössze két állomással rendelkezik majd.

Első lépésként indítsunk el a szövegszerkesztőnket a fentebb olvasható módon! Ekkor egy üres ablakot kapunk, amelybe szép lassan begépeljük az alábbi sorokat, ügyelve arra, hogy minden tökéletesen megegyezzen. Fontos a vesszők, pontok, pontosvesszők, új sorok megléte, helye!

```
With Route
.Comment Ez egy tesztpálya
.Timetable Kálvin tér - Astoria
.Gauge 1435

With Train
.Folder V63 Bhv
.Run(0) 1

With Structure
.Rail(0) MAV-120a/Sin/Egyenes.b3d
.Ground(0) MAV-120a/Fold/Grass.b3d

With Texture
.Background(0) MAV-120a/Hatter/Bluesky2.bmp

With Track
0
.Accuracy 2
.Adhesion 150
.Height 0.5
.Ground 0
.Back 0
.Sta Kálvin tér;8.2940;8.3000;;1;0;;;10;180;;
```

```
100
.Stop 1

350
.Sta Bajcsy-Zs. út;8.3125;T;;-1;0;;;;120;;

450
.Stop -1
```

Ugye nem is volt nehéz bemásolni? Jól jön a copypaste, igaz? :-) Van egy olyan tippem, hogy eddig kínai a dolog, értelmetlen parancsok értelmetlen struktúrában. De sebaj, néhány magyarázat, és minden érthető lesz!

A BVE úgy működik – mint ahogy Bubu azt nagyon helyesen meglátta – mint egy programnyelv. Nekünk csak a keretet kell megadnunk, majd megmondani mikor mit csináljon, az OpenBVE mindezeket végrehajtja gond nélkül.

Ha a fentieket bemásoltuk, mentsük el (Mentés másként → File típusa=minden fájl, Kódolás=UTF-8!) mondjuk *Teszt.csv* néven a Railway/Route könyvtárunkba, oda, ahol a többi pálya is van! Az ablak maradhat nyitva, így könnyebben módosítunk rajta a későbbiekben.

Ezek után indítsuk el az OpenBVE-t és válasszuk ki a Teszt.csv-t, majd menjünk rajta végig! Elvileg gond nélkül meg kellett ennek történnie.

Nem egy nagy durranás, igaz? Valóban. A pálya nyílegyenesen vezet, háttérben valami felhős égbolt, körülöttünk minden sík és szép zöld, fölöttünk felsővezeték amit nem tart semmi, az egyetlen észrevehető különbség egy darab tábla ami a megállás helyét jelöli. DE (történelmünk legfontosabb szava!) Ezt mind mi csináltuk! Igen. Mi monduk meg a programnak, hogy melyik járművet kívánjuk vezetni, mi mondtuk meg neki, hogy nyílegyenes legyen a pálya, mi mondtuk meg neki, hogy hova tegye le azt az egy szem táblát, mi állítottuk be a hátteret, valamint a talaj- és síntípust is. Gondolom ott motoszkál a kérdés, hogy hogyan. A válasz nem lesz rövid, de próbálkozzunk meg vele, vesézzük ki egyesével a leírt sorokat! Íme:

```
With Route
```

Ezt a mondatot csak így le kell írnod, ezzel mondjuk meg a programnak, hogy innentől egy hivatkozásokat tartalmazó rész jön. Ne csinálj vele semmit, viszont fontos, hogy az alant lévő soroknak mindenképpen ez után kell jönniük!

```
.Comment Ez egy tesztpálya
```

Ha figyeltél, amikor kiválasztottad a pályát az OpenBVE-ben, akkor a jobb felső ablakban ezt írja ki a program. Indítsd csak el az OpenBVE-t (igen, MOST!), majd keresd meg a Teszt.csv file-t, és kattints rá egyet! No, szóval ha egyet kattintottál, akkor a jobb felső, *Részletek* ablakban kiírja azt a szöveget, amit a .Comment után írtál. Ez lehet több sor is, a gép bírja, elég hosszan. Úgyhogy most a szövegszerkesztőnkben, a Teszt.csv file-ünkban módosítsuk is, mondjuk úgy, hogy azt írja ki az OpenBVE-ben a Route Overview ablakban: „*Ez a valaha is készített legjobb és legigényesebb tesztpálya*”. Ugye nem lesz gond, egyszerűen töröljük ki a .Comment után az „*Ez egy tesztpálya*” részt, és írjuk be a fentieket. Mentsük el (egyszerűen elég mentést választani), majd indítsuk el az OpenBVE-t, válasszuk ki a Teszt.csv-t, kattintsunk rá egyet és máris látjuk a változást. A játékra nincs kihatással, hogy mit írunk ide, de nem árt hasznos információkkal ellátni a játékost, mivel is fog játszani ha betölti az útvonalat.

Ugorjunk is a következő sorra!

```
.Timetable Kálvin tér - Astoria
```

Ha az OpenBVE-ben megnyomjuk a Ctrl+T-t, akkor előugrik a menetrend. Ennek a legfelső sora azt tartalmazza, amit a *.Timetable* parancs mögé írtunk. Azaz ha azt írjuk ide, hogy „*.Timetable Kutyafüle*”, akkor a menetrendünk fejlécében az fog szerepelni, hogy *Kutyafüle*. Szintén csak információs jelleggel bír.

```
.Gauge 1435
```

Ez a parancs a nyomtávot óhajtja beállítani. Magyarországon a nagyvasúti és a városi vasúti sínek is 1435 milliméteresek, ezért szerepel 1435 a *.Gauge* mögött. Ha kisvasutat szeretnénk építeni, akkor például 760 lenne a helyes érték, ha kisvasútunk 760 milliméteres nyomtávolságú. Az oroszoknál szélesebb a nyomtáv, mint nálunk, így ők ide egy nagyobb értéket írhatnak, ha a saját vasútjukat kívánják felépíteni.

```
With Train
```

Ha van már némi előzetes angoltudásotok, akkor valószínűleg már kitalálhattátok, hogy a *With Route* rész a pálya alapvetéseit tartalmazta, innentől kezdve kezdődik a pályában használt vonatra tartozó parancsszekció.

```
.Folder V63 Bhv
```

Az egyik leggyakrabban használt parancs. Ez mondja meg, hogy a pályán milyen járművel mehetünk. Egy könyvtárnevet írhatunk be ide, méghozzá bármelyiket, ami a *Train* könyvtárunkban van. Tehát ha például a Kisföldalatti szerelvényével kívánunk menni, ami a Train könyvtáron belül az *MFAV* könyvtárban van, akkor módosítsuk *.Folder MFAV* -ra ezt a sort, mentsük el és indítsuk el az OpenBVE-t!

Mondom, indítsd el!

És máris csoda történt, immáron a nyílegyenes vonalunkon egy MFAV-val szaladgálhatunk. Ez bármilyen jármű-pálya kombinációra igaz, akár egy Csörgővel is mehetünk a metróalagútban függetlenül a nyomtávtól, felsővezetéktől, bármitől. Ezzel azt a kérdést is megválaszoltuk, hogy hogyan lehet alapból másik járművel menni egy bizonyos pályán.

```
.Run(0) 1
```

Ezzel a vonat hangját tudjuk beállítani a különböző típusú síneken. Itt ez azt jelenti, hogy a 0-s számmal használt síntípuson való haladás esetén (lentebb majd láthatod, hogy ez az egyenes sín lesz) a jármű könyvtárából a Run1.wav vagy Run1.flac hangot fogja lejátszani. Később erre majd visszatérünk egy másik leckében, egyelőre most ezt kell tudnod erről.

```
With Structure
```

Ahogy a With Train-nél, úgy itt is ez egy új szekció kezdetét jelöli, jelen esetben ebben a részben fogjuk felsorolni mindazon 3D-s objektumokat, amiket a pálya használni fog.

```
.Rail(0) MAV-120a/Sin/Egyenes.b3d
```

Ezzel a paranccsal adjuk meg, hogy a 0-s számú sín típusát, kinézetét a program az Egyenes.b3d file-ból vegye ki. Az útvonalat a *Railway/Object* könyvtártól kell megadni. A későbbiekben itt még

lesznek olyan parancsok, hogy .Rail(1) blablabla meg .Rail(2) blablabla, ezekre tényleg visszatérünk.

```
.Ground(0) MAV-120a/Fold/Grass.b3d
```

Ezzel a fentihez hasonló módon adjuk meg a sínt övező föld típusát, kinézetét, amit a program jelen esetben a Grass.b3d file-ból vesz. Ezt is szabadon módosíthatjuk, most én azt mondanám, hogy ne piszkáljuk, később játszadozunk ezzel is.

```
With Texture
```

Itt ismét egy újabb szekció veszi kezdetét, ezúttal a pályában használt háttérképek textúráit fogjuk összegyűjteni. Valószínűleg egy pálya építésénél ezek megadásával fog a legrövidebb idő eltelni.

```
.Background(0) MAV-120a/Hatter/Bluesky2.bmp
```

Ez megint csak hasonlít az előzőkhöz, annyiban, hogy a háttérképet állítjuk be itt, hogy futás alatt mit mutasson a program a háttérben. Ezt is megváltoztathatjuk egy játékon belül, majd elmondom, hogyan.

Lássuk csak a következő sort!

```
With Track
```

Ezzel mondjuk meg az OpenBVE-nek, hogy innentől jönnek a pályaadatok. Semmi más teendőnk nincs vele, mint ugyanígy leírni. :-)

```
0
100
350
450
```

Ezek mind-mind távolságot jelölnek, méghozzá méterben.

Ha szeretnénk bármit is csinálni a pályával, azaz például azt, hogy legyen egy állomásunk a pályán 300 méternél, akkor beírjuk a sor elejére, hogy *300*, a program pedig úgy értelmezi, hogy mindaz, ami ez után jön, azt 300 méternél kell végrehajtania. De természetesen nem kell egy pályának a 0 méternél elkezdődnie, tehát lehet 15 673.429 méternél (azaz 15 673 429 milliméternél) is a kezdőpont, gondolva arra, hogy később esetleg szeretnénk a pálya elejébe is építeni szakaszokat. Csak rajtunk múlik, mivel kezdjük.

Plusz még az is működik, hogy még csak sorba sem kell tenni ezeket a métereket, tehát teljesen összevissza is beszámozhatjuk a pályát, ezt azonban nem tartom célravezetőnek, úgyhogy maradjunk csak ennél a sorban írásnál.

Miután jól belebonyolítottalak, nézzük csak sörönként, izé, soronként.

```
0
```

Mint az előbb is utaltam rá, ezzel azt mondjuk a programnak, hogy a lentebbi parancsokat a nulla méternél hajtsa végre. Ha még ugyanabban a sorban akarod hozzáírni az adott méterre vonatkozó parancsokat, akkor vesszőt kell a 0 után tenni, és jöhet a hozzárendelt parancs, tehát pl.:

```
0,.Sta Állomásnév […],
```

Viszont ha a hozzárendelt parancsokat új sorokba írod az Enter billentyű használatával, akkor a vessző a 0 után elhagyható.

```
.Accuracy 2
```

A pálya minőségét állíthatjuk be ezzel a paranccsal. Ha a 2-es helyére 0-át írunk, akkor az ultrabrutáljól lefektetett és karbantartott síneket jelent, ha 4-et, akkor az pedig kegyetlenül rosszakat. Tehát itt a kettes azt jelenti, hogy tulajdonképpen jól elmegy rajta a járgányunk, kisebb döccenők előfordulhatnak.

Ha kiderül, hogy mondjuk 100 métertől nagyon rossz a sín a 150. méterig, akkor ezt be tudjuk úgy állítani, hogy beírjuk a file-unkba:

```
100
.Accuracy 4
150
.Accuracy 2
```

Mint látható, a századik méternél átvált a pálya nagyon rossz értékre (amit a négyes jelképez), aztán 150 méternél visszaáll a pálya elején beállított kettesre (ami jobb minőség). Ezzel lehet bátran játszadozni. Mint minden parancsnál, ennél is pontot kell a parancs elé tenni.

```
.Adhesion 150
```

Ezzel a csúszás mértékét állíthatjuk be a pályán, ami függhet például az időjárástól, vagy bármilyen hasonló tényezőtől. A 150 egy teljesen átlagos érték normális időben, ha 100-at írnánk be, akkor jobban csúszna a gép, mintha esne az eső, 85-nél fagyos pályán haladhatunk, 50-nél pedig a hóval borított pályával megegyező tapadásunk van csak. A parancs működése dettó ugyanolyan, mint fentebb a .Accuracy-nak.

```
.Height 0.5
```

A sínek magasságát adja meg a talajhoz képest, méterben. Fontos, hogy nem tizedesvessző van, hanem pont!!!

```
.Ground 0
```

Ezzel adjuk meg, hogy milyen legyen a minket körülvevő terep, azaz milyen színű legyen a föld, melyik típust válassza.

Emlékeztek még? Volt egy olyan rész már a leírásban a *With Structure* után, hogy

```
.Ground(0) MAV-120a/Fold/Grass.b3d
```

amivel megadtuk, hogy a 0-s számhoz a Grass.b3d tartozzon.

No ezzel a .Ground 0, paranccsal azt adjuk meg, hogy ettől a métertől fogva (tehát jelen példánkban a nulladik métertől) a 0-s számhoz hozzárendelt fájlt jelenítse meg mint talaj, azaz a Grass.b3d-t.

Ez is bármikor módosítható, azaz ha megadunk egy másik file-t a With Structure után azzal a

paranccsal, hogy *.Ground(1) valami*, akkor a *.Ground 0*, helyére beírhatjuk azt, hogy *.Ground 1*, és onnantól fogva az lesz a talaj.

Nem biztos, hogy tisztán érthető, de próbáljátok ki!

```
.Back 0
```

Ezzel pedig a háttérképet adhatjuk meg. Hasonlóan tudjuk használni, mint az előzőt. Ugyanúgy a fájlunk elején a *.Background(0)* paranccsal megmondtuk, hogy nullás számmal melyik fájlra óhajtunk hivatkozni, hogy azt jelenítse meg háttérként.

```
.Sta Kálvin tér;8.2940;8.3000;;1;0;;;10;180;;
```

Az egyik legszebb, és talán a legtöbb paraméterrel rendelkező parancs.

Vele mondjuk meg a programnak, hogy állomás következik.

A .Sta parancs után az alábbi értékek jönnek, amiket pontosvesszővel kell elkülöníteni:

1. **Állomás neve** – jelen esetben Kálvin tér. Az OpenBVE ezt írja be a menetrendbe is, úgyhogy nem árt pontosan leírni. Nem célszerű túl hosszú nevet adni, mivel akkor alacsonyabb felbontás esetén nem tudja az egész nevet megjeleníteni a program.
2. **Érkezési idő** – jelen esetben 8.2940 Ezt óó.ppmm formában kell megadni, azaz óra-(pont!)-perc-másodperc. A legelső állomásnál ez azt az időt jelöli, amikor a vezetőülésben helyet foglalunk és készen áll a pályán a járművünk az indulásra. A többi állomásnál ez az érkezési idő, ehhez viszonyítja a kiírásnál a késést. Tehát jelen esetben a szimuláció ideje reggel 8 óra 29 perc 40 másodperckor indul.
3. **Indulási idő** – jelen esetben 8.3000 Ezt is az előző formátumban kell megadni, ekkor indulhat a jármű az állomásról, tehát reggel 8 óra 30 perc 0 másodperckor. A program úgy próbálja becsukni a jármű ajtajait, hogy ebben az időpontban el tudj indulni. Persze ha késel vagy sok az utas, akkor ez nemigen fog összejönni...
4. **Állomási figyelmeztető** – jelen esetben ez az érték üres, tehát láthattok két pontosvesszőt, ami között nincs érték. Üresen hagyva ugyanaz, mintha nulla lenne ott. Ha 1-es számot írunk be, akkor ha nem álltunk meg ezen az állomáson, a program figyelmeztet, hogy meg kellett volna. Ha üresen hagyjuk vagy nullát írunk be, akkor pedig nem figyelmeztet.
5. **Peron beállítása** – jelen esetben 1. Ezzel adhatjuk meg, hogy melyik oldalon van az állomáson a peron, így az OpenBVE azon az oldalon ad ajtónyitási engedélyt, és arról az oldalról jön majd a hang. –1 esetén bal oldali nyitás van, 1 esetén jobb oldali nyitás, 0 esetén nem kell ajtót nyitni, csak meg kell állni és meg kell várni az indulási időt. Ha B betűt írunk be, akkor mindkét oldalon nyílnak majd az ajtók.
6. **Kijárati jelző állása** – jelen esetben 0. Ezzel adhatjuk meg, hogy a kijárati jelző mit mutasson. 0 esetén mindig zöldet mutat (feltéve ha nincs előttünk a pályán másik vonat), 1 esetén addig vöröset mutat amíg meg nem közelítjük a kijárati jelzőt, vagy az állomásról való indulási időig több, mint 15 másodperc van.
7. **Biztonsági rendszer aktiváló** – jelen esetben üresen van hagyva. No ehhez nagyon nem értek, amennyit más oldalakról jól kimazsoláztam, ha 0 van ide beírva, akkor ATS-t aktivál, ha 1, akkor pedig ATC-t, ha semmit sem írsz be, akkor viszont egyiket sem aktiválja. Ez vonatfüggő parancs, tehát ezt a paramétert csak azon vonatoknál veszi figyelembe a program, amelyek az alapértelmezett OpenBveAts beépülőt (plugint) használják biztonsági rendszerként. Erről majd egy másik leírásban olvashattok, egyelőre haladjunk tovább a .Sta parancs paraméterei tisztázásában!
8. **Érkezési hang** – jelen esetben üresen van hagyva. Itt azt állíthatjuk be, hogy milyen

hangot játsszon le az OpenBVE, miután a szerelvény beérkezett az állomásra és az ajtók kinyíltak. Útvonalként nem kell a teljeset megadni, a Railway/Sound könyvtár az alap, innen kell csak megadni úgy, hogy *könyvtárneve/valami.wav*. Csak .wav és .flac formátumú hangokat képes lejátszani a program, erre ügyeljünk!

9. **Állomási állásidő** – jelen esetben 10. Másodpercben mutatja, hogy járművünknek mennyi időt kell minimum állnia egy állomáson, függetlenül attól, hogy késett-e vagy pontos. Ezzel az utasforgalmat is lehet szabályozni, azaz kisebb állomásnál, kevés felszállónál állíthatjuk kis értékre, ahol pedig nagy az utasforgalom, több ideig tart az utascsere, akkor nagyobb számot célszerű ide beírni. Viszont! Az utasszám is befolyásolja az állomáson tartózkodási időt! Azaz ha 10 másodperces tartózkodási időt állítasz be, és az utasok számát megemeled 250-re, akkor biztosan nem fognak bezárulni az ajtók 10 mp után, mert ennyi idő alatt nem tudnak felszállni az utasok.
10. **Utasszám** – jelen esetben 180. Ez a járművünk zsúfoltságát adja meg, azaz mennyire lesz tömeg ha elindulunk ettől az állomástól. 0 esetén egy lélek sincs a járgányon, 255-nél (ez a maximum) meg már dudorodik a jármű oldala, annyira tele van. Hatással van az utasvéleményre is, ugyanis ha nagyon alacsony értéket írunk be, akkor a kevés utas nem tud annyira morgolódni, ha hirtelen nagyon behúzod a féket vagy túl nagy gyorsulással indulsz meg.
11. **Indulási hang** – jelen esetben üresen van hagyva. Az érkezési hanghoz hasonló, annyi a különbség, hogy az ajtók zárása előtt játssza le ezt a hangot. Ugyanúgy kell az elérési útvonalát megadni a .wav vagy .flac fájlnak, mint az érkezési hangnál. Ez az utolsó paramétere a .Sta parancsnak, de nagyon fontos, hogy megfelelő számú pontosvessző legyen, különben belezavarodhat a program!
12. **Menetrend index** – ezzel egy menetrendet mutató képet jeleníthetünk meg az állomásra érkezéskor. Nem túl gyakran használt paraméter, ezért ebben a leírásban nem részletezzük, egyelőre hagyjuk üresen.

```
100
.Stop 1
```

A .Stop paranccsal a megállás helyét adjuk meg. A 100 ugyebár azt mutatja, hogy ez a 100-adik méternél legyen. Az 1-es értékkel azt adjuk meg, hogy a program tegye ki a megállást jelképező táblát (igen, azt a piros, élére állított négyzetet) a sín mellé a jobb oldalra. Ha –1-et írunk ide (tehát .Stop –1), akkor a bal oldalra teszi. Ha 0-t írunk ide, akkor nem teszi ki a táblát. Ez utóbbit akkor célszerű használni, ha másik, saját készítésű megállótáblát rakunk le. De ennek menetét majd később, most jó lesz a –1 vagy az 1 is.

```
350
.Sta Astoria;8.3125;T;;-1;0;;;;120;;
```

Ezt talán már nem kell kiveséznem, lévén megegyezik az előzővel, némi különbségekkel. Ugye ezt először is 350 méterhez tettük le.

Másrészt az állomás neve Astoria, ha már a menetrendben ezt adtuk meg, akkor itt is szerepeljen ez.

Harmadrészt pedig feltűnhet az indulási idő helyetti T betű. Ezzel azt jelképezzük, hogy a járművünk innen nem megy tovább, ez volt a célállomása. Ha pedig olyan állomást akarunk csinálni, ahol a vonatnak nem kell megállnia, hanem áthalad, akkor az érkezési idő helyére P betűt írunk, az indulási idő helyére pedig vagy beírjuk, hogy mikor kell áthaladni az állomáson, vagy üresen hagyjuk, az OpenBVE mindkét megoldást elfogadja.

```
450
.Stop -1
```

Mint fentebb is olvasható, ezzel jelöljük meg, hogy hol legyen a megállás helye az állomáson. Jelen esetben 450 méternél. Ha lefelejted, a menetrendben nem fog megjelenni az állomás neve, hiába van *.Sta* parancsod, de ebben az esetben az OpenBVE hibaüzenetet is generál a pálya elindításakor.

# 2. rész

Első leckénket ott hagytuk abba, hogy építettünk egy rövid szakaszt, amelyen a vonatunk el tud indulni és megállni a következő állomáson. A valós életben azonban meglehetősen kevés útvonal tudhatja magának azt a sajátosságot, hogy nyílegyenesen vezet a semmibe, míg mi csak egy megállót közlekedhetünk rajta. Így mostani leírásomnak az a célja, hogy valamennyi segítséget nyújtsak az útvonal kialakításához.

Mi lehet a legalapvetőbb egy pályán, ami nagyon sokat dob a hangulatán? Igen, az ívek azok. Jobbra és balra, váltakozva, ki-hogy szereti. A dolog könnyebb, mint gondolnánk. Adott ugyebár a pályánk és az elhatározásunk, hogy teszünk be egy jobbos ívet a 200. métertől kezdődően, egészen a 275. méterig, és mondjuk az ív sugara legyen 300 méter. Innentől a megoldás egyszerű, az alábbi parancsokat kell használnunk:

```
200
.Curve 300;0
275
.Curve 0;0
```

Miről is van szó:
A 200. méternél a .Curve paranccsal „üzenem” a programnak, hogy én egy ívet szeretnék a pályába ettől a ponttól, ami 300 méter sugarú (jobbra! Ha bal ívet szeretnénk, akkor negatív értéket kell ide beírnunk, tehát 300 helyett –300-at!) és 0 mm túlemelés legyen rajta (erről később). Tehát a pontosvessző előtti érték az ív sugara, a pontosvessző utáni érték a túlemelés.

A 275. méternél pedig a fent látható paranccsal azt mondom a programnak, hogy innentől kezdve egy 0 méter sugarú ívet tegyen le, azaz gyakorlatilag egyenesen vezesse tovább a pályát. Fontos tudni azonban, hogy mivel a tesztpályához csak 25 méter hosszú sínmodellt választottunk, így hiába írod azt be, hogy *198,.Curve –300* akkor is csak a kétszázadik (200.) métertől fog látványosan kanyarodni a sín, miközben a vonat már idő előtt elmászik a sínről, így egyelőre maradjunk a 25 méterenkénti parancsmegadásoknál.

Valamint nem árt utánanézni, hogy villamoséknál vagy nagyvasúton mik a jellemző ívek. A túl szűk íveket nem szeretik se a járművek, se az utasok, se a program. Így ne is kísérletezzünk 40 méter hosszú 30 méter sugarú ívekkel, mert abból nem jön ki semmi. Alkalmazzunk ilyen esetekben 50 méter hosszot, valamint olyan 60-80 méter sugarú ívet. Látványilag egyáltalán nem zavaró, ráadásul közlekedni is lehet rajta. Az OpenBVE program pályakezelése miatt, azaz hogy eddig 25 méteres részekből építi fel a pályát, most még az általunk épített szűk ívek meglehetősen szögletesek. Ezen egy későbbi leckénk során egy másik programmal segíteni fogunk, most érjük be ezzel!

Túlemelés: ívek esetén alkalmazzák, a külső sínszálat kis mértékben megemelik, ezzel kvázi megdöntve a vonatot. Ennek több oka van, én mindenesetre nem részletezném, lévén nincs meg

hozzá a kellő szaktudásom, hülyeségeket meg nem szeretnék írni. A neten gyanítom lehet találni olyan táblázatokat/írásokat, amelyben szerepel, hogy mekkora túlemelés tartozik milyen sugárhoz és sebességhez. Megemlíteném még, hogy az íveket nem kötelező lezárni a .Curve 0;0 paranccsal, több ív is fűzhető egymásba. Tehát például ha azt szeretnénk, hogy a járművünk egy átmeneti ív segítségével kanyarodjon balra, akkor az alábbiakat írhatjuk be a pályánkba:

```
200
.Curve -800;0
225
.Curve -500;0
250
.Curve -200;0
300
.Curve 0;0
```

Mint azt látjátok, fokozatosan csökkentettem az ív sugarát, így véve egyre élesebbé az ívet. Néha célszerű így használni, főleg nagyvasútnál adhat látványos képet. Ugyanígy akár "S-kanyart" is készíthetünk:

```
400
.Curve 1000;0
500
.Curve -1000;0
600
.Curve 0;0
```

Ha már így benne vagyunk a kanyargásban, akkor nézzük meg, mi van akkor, ha mindezt emelkedőkkel vagy lejtőkkel szeretnénk kombinálni. Természetesen ez is egyszerűen van megoldva a programban, egy próba – na és persze a leírás elolvasása után – mindenki rájön a dolog ízére.

A parancs nem más, mint a *.Pitch*. Használatához nézzük, mit kell beírnunk egy 5 ezrelékes, 100 méter hosszan tartó emelkedőhöz:

```
300,.Pitch 5
400,.Pitch 0
```

Ennyi. Valóban csak ennyi. 300 méternél elkezdődik egy 5 ezrelékes emelkedő, 400 méternél pedig a pálya visszatér a síkba. Ha lejtőt szeretnénk, akkor negatív értéket kell beírnunk. Nagyvasúton a maximális érték 35-40 ezrelék környékén van, annál nagyobb emelkedőnél már fogaskerekű az ajánlatos. Mint az íveknél, itt is célszerű átmeneti szakaszokat létrehozni, tehát a sík terep után ne egy 25 ezrelékes emelkedőt tegyünk be egyből mert rondán néz ki és valótlan is, hanem szép fokozatosan módosítsuk az értéket:

```
275,.Pitch 1.67
300,.Pitch 3.33
325,.Pitch 5
350,.Pitch 6.67
375,.Pitch 8.33
400,.Pitch 10
525,.Pitch 8.33
550,.Pitch 6.67
575,.Pitch 5
600,.Pitch 3.33
625,.Pitch 1.67
650,.Pitch 0
```

Jó, lehet, hogy ez sem teljesen reális, de szerintem érzitek a különbséget.

No, miután mindenki kiszáguldozta magát a pályán, vezessünk be némi sebességkorlátozást!

Egy 40 km/h-s sebességkorlátozást az alábbi módon tudunk megvalósítani az OpenBVE-vel:

```
0,.Limit 40;1
```

Természetesen ez is adott méterhez tehető le, és addig érvényes, amíg egy következő ilyen parancsot le nem tettünk. A *.Limit* után az első paraméter (jelen esetben az 40-es szám) adja meg, hogy mennyi a megengedett legnagyobb sebesség km/h-ban. Ha ezt meghaladjuk több mint 1 km/h-val, akkor a gép jelezni fog. A második paraméter, azaz a 1-es pedig azt mutatja, hogy a pályától jobbra (pozitív érték esetén) vagy balra (negatív érték esetén) mennyi méterre rakjon le a program sebességkorlátozást mutató táblát. Ha eme érték helyére nullát írunk, akkor nem mutatja a korlátozást a program a beépített táblával, hanem nekünk kell készítenünk egyet és .FreeObj-ként betenni. De erről majd később. Most mindenesetre tegyünk egyet a pályánkba!

Ha kiszórakoztátok magatokat a szemetebbnél-szemetebb sebességkorlátozások kitűzésével, akkor tegyünk be még egy vágányt a már meglévő mellé. Jó ötlet, nem? A megvalósítás egyszerű, ehhez a *.Railstart* parancsot kell használnunk:

```
0,.Railstart 1;-4;0;0
```

Ezzel azt mondtuk a programnak, hogy már a pálya indulásakor legyen egy második vágány a mi vágányunktól négy méterre balra, vele egy szintben, ugyanolyan síntípussal. De lássuk az értékeket részletezve is, egymás után!

- **0,** – Ezzel ugye azt adtuk meg, hogy 0 métertől kezdődjön a második sínszál
- **.Railstart** – Ez maga a parancs
- **1;** – Ezzel azt adjuk meg, hogy mi legyen az új pálya hivatkozási száma. Ugyebár az általunk használt sín mindig a 0-s számot viseli, így ennek most az 1-est adjuk, de lehetne bármilyen más nullánál nagyobb egész szám. Ez majd olyan parancsoknál lesz fontos, ahol a sínpálya index-számára hivatkozunk.
- **–4;** – Ezzel azt adjuk meg, hogy mennyi méterre és milyen irányban helyezkedjen el jobbra (pozitív érték esetén) vagy balra (negatív érték esetén) a mi vágányunktól a pluszban lefektetni kívánt sín. Mivel az érték –4, azért balra, 4 méterre lesz.
- **0;** – Ezzel azt adjuk meg, hogy mennyi méterre legyen felfele (pozitív érték esetén) vagy lefele (negatív érték esetén) a mi vágányunktól függőleges irányban a pluszban lefektetni kívánt sín. Mivel az érték itt nulla, ezért egy szintben lesznek.
- **0** – Ezzel azt adjuk meg, hogy milyen síntípust vegyen fel az új vágány. Mivel a .csv file-unk elején csak egy síntípust, az Egyenes.b3d-t számoztuk be (ugyebár ő kapta a 0-s számot a .Rail(0) parancsnál), ezért ezzel azt mutatjuk, hogy ez a sín is ugyanezt használja.

A fenti értékek természetesen majd a pályán később változtathatók, de most ne menjünk ebbe bele, szeretném, ha ezt a részt megértenétek.

Mivel létrehoztunk egy új sínt, talán nem lesz végig rá szükségünk, ezért zárjuk is le egy másik paranccsal, ami nem lesz más, mint a *.Railend*.

Használata egyszerű, ha 500 méternél be akarjuk fejezni az előbb nyitott, 1-es számúnak elkeresztelt vágányt, akkor az alábbiakat kell beírnunk az útvonal file-unkba:

```
500
.Railend 1
```

Az egyetlen egy értékkel (ugyebár az 1-essel) mondhatjuk meg a programnak, hogy melyik sínszálat óhajtjuk megszüntetni. Igen, mindössze ennyit kell beírnunk.

Most pedig tessenek nekiállni gyakorolni, mindjárt jön a házi feladat! Mu-ha-ha-ha! :-)

Eredeti leírás: IST, 2006. január 14-18. (1., 2., 3. rész)
Második kiadás: Gothpaladinus, 2013. január 18.
Lektorálás: Phonteus Nevolius

www.bveklub.hu