Programozói dokumentáció

3d Space Game

A program futtatásához szükség van az SDL2 könyvtárra. A program C99 nyelven készült.

Modulok:

- **main:** A fő programmodul. Inicializálja az SDL ablakot és SDL kirajzolással kapcsolatos funkciókat tartalmaz. Meghívja a játékban szereplő objektumok inicializáló függvényeit. Beolvassa a modelleket. Tartalmazza a játék fő ciklusát. Tartalmazza a menürendszerrel kapcsolatos funkciókat és a fájlkezelést. Itt található a billentyűzet bemenetének feldolgozása is.
- **3d_muveletek:** Főként a térleképezéssel kapcsolatos függvények modulja. Tartamaz vektor és mátrix műveleteket. Itt foglalnak helyet a testek térbeli fizikáját megvalósító függvények. Az objektum és fizika inicializáló függvények is itt vannak.
- **kirajzolas:** Grafikai megjelenítést végző funkciók.
- **dinamikus_mezok:** modellek dinamikus memóriában való tárolását végző segédfunkciók.
- **3d_tipusok:** Csak header állomány. A program által használt struktúrákat és enumerációkat gyűjti össze.
- **menu:** Csak header állomány. Adatstruktúrák a menü létrehozásához.

Forrás mappák:

- **fontok:** ttf betűket tartalmazó mappa. (A játékban ezzel a betűtípussal vannak kiírva a felíratok.)
- **hangok:** A játék zenéjét és hangeffektusait tartalmazza.
- **kepek:** A menü hátterében látható képet tartalmazza.
- **modellek:** A játékban szereplő 3d-s modelleket tartalmazza.

Main modul függvényei:

- void **kamera_mozgatas** (kamera *kam, kontroll kont, double delta_t)
- void **loves** (objektum_tomb *lovedekek, vek3d poz, vek3d irany, Mix_Chunk *hangok)
  Lovedék objektum pozíciójának a kilövő űrhajóhoz állítása és nézőirányú sebesség adás.
- bool **meret_allitas_vizsgalat** (bool allit)
- void **ablakmeret_allitas** (SDL_Window *window, SDL_Renderer *renderer)
  Vált az ablak méretek közt.
- mesh **mesh_beolvasas** (char *fajlnev)
  Beolvas egy 3D modellt a fájlból.
- void **gyorsmenu_valto** (menu_irany irany, gyorsmenu_allapotok *allapot)
  Lépked a gyorsmenü menüpontjai közt.
- void **fomenu_valto** (menu_irany irany, fomenu_allapotok *allapot)
  Lépked a főmenü menüpontjai közt.
- void **beallitasok_valto** (menu_irany irany, beallitasok_allapotok *allapot)
  Lépked a beállítások menüpontjai közt.
- void **kilepes_elott_valto** (menu_irany irany, kilepes_elott_allapotok *allapot)
  Lépked a kilepes előtti menü menüpontjai közt.
- void **jatek_kezdes_valto** (menu_irany irany, jatek_kezdes_allapotok *allapot)
  Lépked az egyjátékos/kétjátékos mód választó menü menüpontjai közt.

- int **elfogadas** (jatek_allapotok *jallapot, fomenu_allapotok *fomenu_allapot, jatek_kezdes_allapotok *jatek_kezdes_allapot, gyorsmenu_allapotok *gym_allapot, beallitasok_allapotok *beall_allapot, kilepes_elott_allapotok *kilepes_elott_allapot, int *hany_jatekos, SDL_Window *window)
  Az aktuálisan fókuszban álló menüponthoz rendelt műveletet hajtja végre.
- int **billentyu_reakcio** (kontroll *kont, int *hany_jatekos, bool *megsemmisult, kamera *kam, objektum_tomb *objektumok, objektum_tomb *lovedekek, jatek_allapotok *jatek_allapot, fomenu_allapotok *fomenu_allapot, jatek_kezdes_allapotok *jatek_kezdes_allapot, gyorsmenu_allapotok *gymenu_allapot, beallitasok_allapotok *beall_allapot, kilepes_elott_allapotok *kilepes_elott_allapot, double delta_t, SDL_Window *window, Mix_Chunk *hangok)
  A lenyomott billentyűkhöz rendelt műveletek meghívása.
- int **billentyu_lekerdezes** (kontroll *kont1, kontroll *kont2, kontroll *mar_lenyomott1, kontroll *mar_lenyomott2, bool *debug_mod, bool *sugo_mod, int hany_jatekos)
  Ellenőrzi, melyik billentyűk lettek lenyomva vagy felengedve.
- void **atmenet_kirajzol** (SDL_Renderer *renderer, int time)
  Fekete kép kirajzolása. (menük váltásakor)
- void **fps_kiir** (SDL_Renderer *renderer, unsigned int *rajzolas_szam, unsigned int *elozo_frissites)
  Kiírja a képernyőfrissítés frekvenciáját.
- void **szog_sebesseg_modositas** (ang3d *szog_sebesseg, double delta_t)
  Segédfunkció a kamera mozgatás funkcióhoz. Korlátozza a szogsebességet, hogy ne érjen el túl nagy értéket.
- void **sebesseg_modositas** (vek3d *sebesseg, double delta_t)
  Segédfunkció a kamera mozgatás funkcióhoz. Korlátozza a sebességet, hogy ne érjen el túl nagy értéket.
- int **inicializalas** (objektum_tomb *objektumok, objektum_tomb *lovedekek, objektum_tomb *reszecskek, mesh_tomb *meshek, kamera *kam1, kamera *kam2, kontroll *kontroll1, kontroll *kontroll2, bool *megsemmisult1, bool *megsemmisult2, int alap_obj_szama, int lovedekek_szama, int meteorok_szama, int reszecskek_szama)
  Inicializálja a játék objektumait a játék megkezdése előtt.
- int **main** (int argc, char *argv[])
  Inicializálja az SDL ablakot, elvégzi a fájlok beolvasását (modellek, betütípusok, képek), inicializálja a változókat, majd belép a játék főciklusába. Itt vizsgálja a billentyűzetről érkező bemenetet, ennek megfelelő műveleteket hív meg, meghívja az objektumok fizikai szimulációját és meghívja az összes kirajzolást. A program leállítása előtt felszabadítja a foglalt memóriát.

3d_muveletek modul függvényei:

- vek3d **matrix_szorzo** (vek3d in, matrix4x4 mx)
  Vektor és mátrix szorzata.
- vek3d **randomizalt_vektor** (int offset, int tartomany)
  Tetszőleges tartományban generál vektort.
- vek3d **vektor_forgatas** (vek3d vek, ang3d szog)
  x, y, és z irányú tengely mentén forgat vektort.
- matrix4x4 **mutato_matrix** (vek3d *pozv, vek3d *celv, vek3d *felv)
  Segédmátrix a pontok kamera szemszögéből nézéséhez.

- matrix4x4 **specialis_matrix_inverz** (matrix4x4 *m)
  A mutató mátrix inverzét képzi, ami a nézet mátrix lesz. (Ez végzi ténylegesen a pontok kameranézetbe mozgatását).
- vek3d **vektor_osszeadas** (vek3d *v1, vek3d *v2)
  Két vektor összege.
- vek3d **vektor_kivonas** (vek3d *v1, vek3d *v2)
  Két vektor különbsége.
- vek3d **vektor_szorzas** (vek3d *v1, vek3d *v2)
  Két vektor szorata koordinátánként.
- vek3d **normal_vektor** (vek3d *iranyv1, vek3d *iranyv2)
  Sík normálvektora két irányvektorból.
- vek3d **vektor_skalarral_szorzas** (vek3d *vek, double skal)
  Vektor és skalár szorzata.
- double **skalar_szorzat** (vek3d *v1, vek3d *v2)
  Két vektor skaláris szorzata
- vek3d **vektorialis_szorzat** (vek3d *v1, vek3d *v2)
  Két vektor vektoriális szorzata.
- double **vektor_hossz** (vek3d *vek)
  Vektor hosszával tér vissza.
- vek3d **vektor_normalizacio** (vek3d vek)
  Egység vektort készít a kapott vektorból.
- void **matrix_nullazas** (matrix4x4 *matrix)
  Kinullaz egy matrixot (inicializáláshoz).
- void **projekcio_matrix_generator** (matrix4x4 *matrix, double *aspect_ratio, double *fov_tangens, double *far, double *near)
- void **z_forgatas_matrix_generator** (matrix4x4 *matrix, double *szog)
- void **y_forgatas_matrix_generator** (matrix4x4 *matrix, double *szog)
- void **x_forgatas_matrix_generator** (matrix4x4 *matrix, double *szog)
- matrix4x4 **nezet_matrix_generator** (kamera *kam)
- vek3d **sik_metszes** (vek3d sik_pont, vek3d sik_normal, vek3d vonal_kezdet, vek3d vonal_veg)
  Metszéspontal tér vissza. Itt metszi a két végpontal adott szakasz a pontal és normalvektorral adott síkot.
- double **siktol_tavolsag** (vek3d sik_pont, vek3d sik_normal, vek3d p)
  Egy pont és egy pontal és normal vektorral adott sík távolságával tér vissza. Ha a visszatérési érték negatív, akkor a pont a sík “túloldalán” helyezkedik el. (A normálvektor iránya határozza meg, hogy melyik a pozitív térfél.)
- int **clip** (vek3d *sik_pont, vek3d *sik_normal, poli *be_haromsz, poli *ki_haromsz1, poli *ki_haromsz2)
  Optimalizáló függvény. A (ponttal és normál vetorral adott) síkba lógó poligont a metszésvonal mentén elvágja. Ennek megfelelően létrehoz nulla, egy vagy két új poligont, amelyek az új alakzatot alkotják. Visszatérési értéke a létrehozott poligonok száma. Több eset lehetséges. Ha teljes kapott poligon a sík pozitív oldalán van, akkor nem kell semmit csinálni. Ha a teljes poligon a sík túlodalán van, akkor kidobja az egészet. => Nem generál új poligont. Ha csak a poligon egy csúcsa lóg a sík túlodlalára, akkor az új alakzat egy négyszög lesz, amit két poligon visszaadásával ad meg a függvény. Ha a poligon két pontja lóg a sík túloldalára, akkor egy poligon visszaadásával leírható, az új alakzat.
- matrix4x4 **rotacio_matrix_tetszoleges_tengely** (float szog_sin, float szog_cos, vek3d iranyv)
  Tetszőleges tengely szerint tetszőleges szöggel forgató mátrixot hoz létre.

- int **haromszog_lekepezes** (poli *bemenet, vek3d *pozicio, bool *vilagit, matrix4x4 *rotacio_x, matrix4x4 *rotacio_y, matrix4x4 *rotacio_z, matrix4x4 *projekcio, kamera *kam, int p_kep_szelesseg, int p_kep_magassag, poli *kimenet1, poli *kimenet2)
  poligont képez le a3d grafika kirajzolásához.
- vek3d **minimap_lekepezes** (vek3d pozicio, matrix4x4 *projekciom, kamera kam, int p_kep_szelesseg, int p_kep_magassag)
  A miitérképen megjelenítendő objektum pozíciók leképezése.
- vek3d **hatter_lekepezes** (vek3d pozicio, matrix4x4 *projekciom, kamera kam, int p_kep_szelesseg, int p_kep_magassag)
  A csillagok és Nap háttérbe vetítéséhez szükséges leképezés.
- int **osszehasonlit** (const void *ph1, const void *ph2)
  Segédfüggvény a “qsort()” standard rendezőalgoritmus paraméterezéséhez.
- fizika **fizika_letrehozo** (double tomeg)
  Inicializálja egy objektum fizikai tulajdonságait.
- void **objektum_letrehozo** (objektum *obj, mesh_tomb *meshek, rgba szin, obj_tipus tipus)
  Inicializál egy játék objektumot. A kapott típusnak megfelelően.
- bool **utkozes** (objektum *obj1, objektum *obj2)
  Egyszerűsített gömb modell szerint vizsgálja, hogy két test ütközésben van-e.
- void **mozgas** (objektum *obj, int t)
  A sebesség alapján változtatja a testek pozícióját és szögelfordulását.
- void **mesh_atszinez** (mesh *m, rgba szin)
  Egy modell összes poligonját a kapott színűre változtatja.
- void **reszecske_effektus** (vek3d poz, rgba kapott_szin, objektum_tomb *reszecskek)
  10 db részecske objektumot lő ki a kapott pozíciótól véletlenszerűen irányba.
- void **fizikai_muveletek** (objektum_tomb *objektumok, objektum_tomb *reszecskek, Mix_Chunk *robbanas_hang, int t)
  Meghívja az összes fizikai törvényt szimuláló függvényt, így a mozgatást és az ütközést is.
- int **mesh_feldolgozas** (objektum_tomb *objektumok, objektum *letiltott, kamera *kam, matrix4x4 *projekcio, int kep_szelesseg, int kep_magassag, int *offset, poli *kirajzolasra_varo)
  Egy mesh (3d modell) összes poligonját síkra vetíti és csökkenő sorrendbe rendezi azokat a nézőponttól mért távolságuk szerint (z tengely).

Kirajzolas modul függvényei:

- int **kep_szelesseg_ker** ()
  Visszaadja a képszélesség változó értékét
- int **kep_magassag_ker** ()
  Visszaadja a képmagasság változó értékét
- void **kep_szelesseg_beallit** (int uj)
  Beállítja a képszélesség.
- void **kep_magassag_beallit** (int uj)
  Beállítja a képmagasságot.
- void **font_megnyitas** ()
- void **font_felszabaditas** ()
- SDL_Texture * **kepernyo_letrehoz** (SDL_Renderer *renderer, SDL_Point meret)
  Létrehoz egy bitmap textúrát.

- void **kiir_vegrehajt** (SDL_Renderer *renderer, SDL_Point pozicio, char *szoveg, SDL_Color szin, TTF_Font *font)
  Adott fontal való szövegkiírás.
- void **kiir** (SDL_Renderer *renderer, SDL_Point pozicio, char *szoveg, SDL_Color szin)
  Wraper a normál méretű szövegkiíráshoz.
- void **kiirOrias** (SDL_Renderer *renderer, SDL_Point pozicio, char *szoveg, SDL_Colorszin)
  Wraper a nagy méretű szövegkiíráshoz.
- SDL_Rect **kepkoordinata_szamol** (SDL_Rect src_rect)
  A kirajzolandó kép az ablak közepére rendezi.
- void **menu_parameter** (bool feltetel, SDL_Point *p_pozicio, SDL_Color *pszin)
  Segédfunkció a menü kiírásához.
- void **fomenu_kiir** (SDL_Renderer *renderer, fomenu_allapotok *allapot, SDL_Texture *img, SDL_Rect texr)
  Kiírja a főmenüt.
- void **gyorsmenu_kiir** (SDL_Renderer *renderer, gyorsmenu_allapotok *gym_allapot)
  Kiírja a gyorsmenüt.
- void **jatek_kezdes_kiir** (SDL_Renderer *renderer, jatek_kezdes_allapotok *allapot)
  Kiírja az egyjátékos/kétjátékos mód választó menüt.
- void **kilepes_elott_kiir** (SDL_Renderer *renderer, kilepes_elott_allapotok *allapot)
- int **hangero_allitas** (bool allit)
  Kiírja a kilépést megerősítő menüt.
- void **beallitasok_kiir** (SDL_Renderer *renderer, beallitasok_allapotok *allapot)
  Kiírja a beállításokat.
- void **menu_kiir** (SDL_Renderer *renderer, jatek_allapotok *jallapot, fomenu_allapotok *fomenu_allapot, beallitasok_allapotok *beall_allapot, jatek_kezdes_allapotok *jatek_kezdes_allapot, gyorsmenu_allapotok *gym_allapot, kilepes_elott_allapotok *kilepes_elotti_allapot, SDL_Texture *img, SDL_Rect texr)
  Az aktuális menü kirajzolását hívja meg.
- void **intro_kirajzol** (SDL_Renderer *renderer, int time)
  Kirajzolja a program indítása után megjelenő Cím felíratot.
- vek3d **jatek_hatter_kirajzol** (SDL_Renderer *renderer, kamera kam, vek3d *csillagok)
  Kirajzolja a csillagokat és a Napot a háttérbe.
- void **haromszog_rajz** (SDL_Renderer *renderer, poli triangle, bool debug_mod, int offset)
  Polygont rajzol ki.
- void **miniterkep** (SDL_Renderer *renderer, int offs_x, int offs_y, int kep_x, int kep_y, objektum_tomb *objektumok, objektum *letiltott, kamera *kam)
  Kirajzolja a Minitérképet, és a HUD képernyő egyébb elemeit.
- void **grafika_kirajzolas** (SDL_Renderer *renderer, SDL_Texture *kepernyo, int offs_x, int offs_y, int kep_x, int kep_y, kamera *kam, objektum *sajat, vek3d *csillagok, int *kir_varo_db, objektum_tomb *objektumok, matrix4x4 *projekcio, poli *kirajzolasra_varo, bool debug_mod, bool megsemmisult)
  A játék 3d grafikáját kirajzoló funkció.
- SDL_Texture * **sugo_letrehoz** (SDL_Renderer *renderer, SDL_Point meret)
  Létrehoz egy textúrát, amire ráírja az irányítás segítséget.
- void **sugo_kirajzol** (SDL_Renderer *renderer, SDL_Texture *sugo)
  Kirajzolja a segítséget.
- void **elvalaszto_kirajzol** (SDL_Renderer *renderer)
  Egy vízszintes vonalat rajzol, amely kétjátékos módban segít elkülöníteni a két képernyőrészt.

dinamikus_mezok modul függvényei:

- void **felszabadit** (void *elemek)
  Bármilyen típusú dinamikusan foglalt memóriát felszabadít.
- vek3d_tomb **vek3d_tomb_letrehoz** ()
  Inline funkció, ami üres vek3d_tomb-öt ad vissza.
- mesh **mesh_letrehoz** ()
  Inline funkció, ami üres meshet ad vissza.
- mesh_tomb **mesh_tomb_letrehoz** ()
  Inline funkció, ami üres mesh_tomb-öt ad vissza.
- void **vek3d_tomb_hozzaad** (vek3d_tomb *m, vek3d uj)
  Inline funkció, amely hozzáad egy paraméterként kapott vektort a paraméterként kapott tömbhöz.
- void **mesh_hozzaad** (mesh *m, poli uj)
  Inline funkció, amely hozzáad egy paraméterként kapott poligont a paraméterként kapott meshhez.
- void **mesh_tomb_hozzaad** (mesh_tomb *m, mesh uj)
  Inline funkció, amely hozzáad egy paraméterként kapott mesht a paraméterként kapott mesh tömbhöz.

3d_tipusok.h adatszerkezetei:

- struct **ang3d:** három tengely szerinti szögeket tartalmaz.
- struct **vek3d:** x,y,z koordinátákat tartalmaz.
- struct **rgba:** szín átadásra való.
- struct **poli:** három vektort és a poligon megvilágításához szükséges színt és fényességet tartalmazza.
- struct **vek3d_tomb:** Vektorokat tároló tömb. Tartalmazza, hogy mennyi elemet tárol és hogy mennyi hely van lefoglalva.
- struct **mesh:** Poligonokat tároló tömb. Tartalmazza, hogy mennyi elemet tárol és hogy mennyi hely van lefoglalva.
- struct **mesh_tomb:** Mesheket tároló tömb. Tartalmazza, hogy mennyi elemet tárol és hogy mennyi hely van lefoglalva.
- struct **lekepezes_valtozok:** A projekció mátrix létrehozásához szükséges adatok.
- struct **kamera:** A kamera pozícióját, néző irányát, mozgási és forgási sebességét tárolja.
- struct **fizika:** testek fizikai jellemzőit tárolja.
- struct **kontroll:** A billentyű lenyomások tárolására létrehozott logikai változók.
- enum **obj_tipus:** Az objektumok identifikálásához szükséges típusok felsorolása.
- struct **objektum:** Egy játékobjektum összes tulajdonsága {

  bool **lathato:** megjelenik az objektum. Ha ez hamis az ekvivalens azzal, hogy a test nincs is jelen a játéktérben.
  bool **fantom:** A igaz fantom értékű objektumok nem ütköznek más objektumokkal.
  int **elet:** Ha eléri a nullát, akkor megsemmisül az objektum.
  int **letezes_ideje:** Korlátozott ideig létező objektumoknál van szerepe. (lövés, részecske effektus)
  bool **vilagit:** Az ilyen objetum a Fényforrástól elfordulva is teljes világosságban marad.
  obj_tipus **tipus:** Az objektum identifikálására szolgál.
  mesh ***pmesh:** Az objektum modellje.
  rgba **szin:** Az objektum színe. Valójában nem ezt használja a kirajzoló függvény, hanem a poligonokban elmentett színt.
  fizika **fizika:** Az objektum fizikai tulajdonságai.

double **sugar:** Az objektum ütközésének vizsgálatához szükséges sugár.
vek3d **pozicio:** Az objektum térbeli elhelyezkedése.
ang3d **szog:** Az objektum térbeli elfordulása.
struct **objektum_tomb:** Az objektumokat tároló tömb.
struct **matrix4x4:** 4x4-es mátrix}

menu.h adatszerkezetei:

enum **jatek_allapotok:** A játék fő állapotait különbözteti meg.

enum **kilepes_elott_allapotok**

enum **jatek_kezdes_allapotok**

enum **beallitasok_allapotok**

enum **gyorsmenu_allapotok**

enum **fomenu_allapotok**

enum **menu_irany**

struct **gy_menupont**

struct **menu_elem:** Egy menüpont és a hozzá tartozó, kiírandó szöveg együttese.

Felhasznált források

- Ötletek és 3D-s megjelenítés koncepciója:
  One Lone Coder: https://www.onelonecoder.com/
- Betütipusok forrása:
  manrope-medium: https://github.com/sharanda/manrope SIL OPEN FONT LICENSE
- SDL könyvtár használata:
  SDL Wiki: https://wiki.libsdl.org/

Felhasznált szoftverek

- Program tervezés:
  Code::Blocks: http://www.codeblocks.org/
- Hang tervezés:
  REAPER: https://www.reaper.fm/
  Audacity: https://www.audacityteam.org/
  Aalto: https://madronalabs.com/products/aalto
- Grafika tervezés:
  Krita: https://krita.org/en/
  Gimp: https://www.gimp.org/
  Blender: https://www.blender.org/

A játéban szereplő audiovizuális elemek saját szellemi tulajdont képeznek.
Zene elérhető: https://soundcloud.com/flying-piano-842623231/space-game-title