Nagy házi

Space invaders – dokumentációk

Felhasználói:

A játékprogram az ismert space invaders játékhoz hasonló játékot valósít meg. A program elindításakor megnyíló ablakban először egy menü jelenik meg, mely a játékos nevéből és két gombból áll, ezek a „new game” és „scoreboard” gombok. A kiírt névre kattintva azt lehet szerkeszteni, de nem írható be tíz karakternél több, illetve amennyiben szóköz kerül a szerkesztő mezőbe, az enter lenyomása (azaz a szerkesztés befejezése és véglegesítése) után azok eltűnnek, a végeredmény egy szóközöktől mentes név. A „scoreboard” gombra kattintva a bal egérgombbal előjön a legjobb 5 pontszámot elért játékos nevét és pontszámát listázó nézet, ahova az adatokat a scores.txt fileból olvassa be a program. Amennyiben ez a file nem létezik, az első indításkor létrehozza a program. A nevek és pontszámok mentése a fileban <név><szóköz><pontszám><újsor> formátumban történnek, ha mégsem, akkor a program hibásan olvashatja be azokat, és valószínűleg nem a megfelelő adatokat írja majd ki. Innen a menübe lehet visszatérni a megjelenő „menu” gombra kattintva a bal egérgombbal. A menüben a „new game” gombra kattintás után kiválasztható a nehézségi szint a megjelenő „easy”, „normal” és „hard” gombokra kattintással, ezután egyből indul is a játék. A nehezebb játékmódokban több űrlény jelenik meg, gyorsabban mozognak és lövedékeik is gyorsabbak. Az űrhajót mozgathatja a játékos a balra és a jobbra nyilakkal, a nehézségi szintnek megfelelően az űrhajó is gyorsabban(/lassabban) mozog. A játékos a space lenyomásával illetve nyomva tartásával lőhet. Az űrlények másodpercenként lőnek, de közülük egyszerre csak egy véletlenszerűen kiválasztott. Az elindított játéknak akkor lehet vége, ha a játékos sikeresen lelőtte az összes űrlényt, az űrlények lelőtték a játékost, vagy pedig elérték a játékos űrhajójának sorát. A játék vége nézet kiírja az elért pontszámot, és amennyiben az a legjobb 5 közé esik, a program menti a scores.txt-be a már ismert formátumban. (Így értelemszerűen az 5. legjobb pontszám kikerül a fileból, és az elért pontszámnál alacsonyabbak a fileban egy sorral lejjebb csúsznak.) Ebből a nézetből is a „menu” gombra kattintással hozható elő újból a menü. A játékprogramból való kilépéshez az ablak jobb felső sarkában megjelenő piros x-re kell kattintani a megszokott módon.

Programozói:

A program legtöbb funkcióját egy eseményhurok valósítja meg, amely az „game” struktúra „int state” változójának értékeit vizsgálja. Ennek megfelelően a következő esetek vannak: 1 – menü, 2 – nehézség kiválasztása, 3 – könnyű játék, 4 - normál játék, 5 – nehéz játék, 6 – játék vége, 7 – legjobb pontszámok. Ha a state változó 0, leáll az eseményhurok, és tulajdonképpen a program is. A program futásának bármely pontján ki lehet lépni a létrehozott ablak piros x gombjával (ez 0-ra állítja a statet).

Típusok és állandók

types.h

A *Projectile* struktúra a játékon belüli lövedékek adatait tárolja, legyen az a játékos vagy az űrlények lövedéke. SDL_Rect típusú struktúrában tárolja a lövedék koordinátáit, valamint egy bool változóból olvasható ki, hogy az adott lövedék éppen ki van-e lőve, még a pályán van, illetve nem talált még el semmit. Egy int változóban még ezen felül eltárolja a lövedék az alapértelmezett y koordinátáját, ahonnan minden lövéskor elindul, ha szükséges.

A *Player* típusú struktúra tartalmazza a játékos legfontosabb adatait, így a nevét, egy játék alatt szerzett pontjait, az űrhajójának irányítására szolgáló bool változókat, melyek a balra és jobbra nyilak, valamint a space lenyomására kell igazra kiértékelődjenek (elengedésre hamisra), az űrhajó koordinátáit és méretét egy SDL_Rect típusú struktúrában, valamint még egy bool változót, amely igaz, amíg a játékos űrhajója „életben van". Ezeken felül a játékoshoz tartozó lézer adatait is tárolja egy Projectile struktúrában.

Az *An_alien* típusú struktúra szolgál a játék alatt megjelenő űrlények egyike adatainak tárolására, illetve ez egy láncolt lista eleme is. SDL_Rect típusú struktúrában az űrlény koordinátáit, egy boolban az állapotát, hogy életben van-e, valamint az űrlények láncolt listájának következő elemét.

Az *Alien_swarm* struktúrába kerül az űrlények láncolt listájának első elemére mutató pointer, két int változóba a sorokban és az oszlopokban lévő űrlények száma (x*y db űrlény kerül a pályára), egy bool menti, hogy létre van-e hozva az űrlények láncolt listája, azaz fel kell-e még majd szabadítani azt, további int változókban az életben lévő űrlények számát, az űrlények mozgásállapotát, mozgásuk irányát, sebességét, az oda-vissza megtett távok számát és a lövés állapotát tárolhatjuk. A mozgás- és lövésállapot egy megfelelő értéke esetén kell mozogniuk vagy lőniük. Ezen felül két Projectile struktúrát ment, (mert egyszerre két lövedéke lehet az űrlényeknek a pályán,) és ebbe az űrlények lövedékeinek adatait, végül egy bool változóban, hogy az „űrlények nyertek-e".

A *Game* struktúrában található a már említett state változó, két másik int változó menti a játékos által kiválasztott nehézségi szint alapján szükséges sebességet és a pontszám szorzóját, mellyel a játék végén a játékos elért pontszáma beszorszandó, és két bool változó pedig a végleges pontszám kiszámítása közbeni állapotokról tájékoztat, azaz megvan-e a végleges elért pontszám, illetve ez össze lett-e hasonlítva a mentett dicsőséglista pontszámaival.

A *Piece* felsorolt típus a forráskép elemeinek – a spriteok – kirajzolását hivatott megkönnyíteni, illetve látható belőle a forráskép összes eleme.

A fenti típusokon kívül vannak még állandók, melyek megnevesítenek pár fontosabb értéket. Ezek a játékablak és a spriteok méretei, a nehézségi szintekhez tartozó pontszorzók és sebességek.

sdl_init.h

A *Colors* struktúra tárolja a játékhoz használt zöld és fekete színeket egy-egy SDL_Color struktúrában.

A *Data_sdl* struktúra menti a különböző SDL függvényekhez szükséges, az ablakra, a rendererre, a fontra, a texturere és az eventre mutató pointereket és a színek Colors struktúráját.

Függvények

main.c

A *main* beállítja a szükséges változókat, létrehoz egy időzítőt, mely másodpercenként 50 SDL_USEREVENT-et generál, beállítja a véletlenszám generátort (srand((int)time(NULL))) megnézi, hogy létezik-e a scores.txt, és ha nem, akkor létrehozza „- 0” sorokkal kitöltve, majd belép az eseményhurokba. Utóbbiból kilépés után törli a betöltött textúrát, fontot, a timert, a renderert és az ablakot, majd kilép az SDL-ből.

types.c

A *set_alien* függvény létrehoz egy An_alien struktúrájú elemet az űrlények láncolt listájából élő állapotban, és a paraméterként kapott x, y, w és h értékeket elmenti az adott űrlény koordinátáit tároló SDL_Rect struktúrába. Sikertelen memóriafoglalás esetén kiír egy hibaüzenetet. Visszatér a paraméterként kapott láncolt lista elejével (ha NULL pointert kapott, akkor az új elem a lista eleje).

A *clear_aliens* függvény felszabadítja az űrlények láncolt listáját, melynek elejét paraméterként kapja meg, majd visszatér egy NULL pointerrel.

A *clear_dead_alien* függvény végignézi a paraméterként kapott láncolt listát, hogy van-e benne halott űrlény. Ha talál, akkor az azt megelőző listaelem következő elemre mutató pointerét átállítja a következő listaelemre, és felszabadítja a „halott” elemet. A lista elejére mutató pointerrel tér vissza, ha nem NULL pointert kapott, különben NULL pointerrel.

sdl_init.c

Az *sdl_init* függvény rendre inicializálja az SDL-t, létrehoz egy int width és int height paraméterek által megadott méretű ablakot, egy renderert és betölt egy fontot és egy forrásképet, valamint ezeket hozzárendeli a megfelelő paraméterként kapott pointerhez. Ha ezek közül valamelyik művelet nem sikerült, arról hibaüzenetet ad vissza és kilép. (InfoC oldalról)

Az *idozit* függvény segítségével hozható létre SDL timer. (InfoC oldalról)

displayers.c

A *draw_piece* függvény kirajzol egy bábút; a forrás a betöltött png, a cel nevű képre rajzol. Hogy melyik bábút, azt a paraméterként kapott Piece felsorolt típus elem, hogy milyen koordinátákra rajzolja, azt a szintén paraméterként kapott x, y, w és h értékek határozzák meg.

A *write_text* függvény kiír egy sztringet a paraméterként kapott színben a kapott fonttal a kapott x és y koordinátájú helyre.

A *draw_menu* függvény kirajzolja a menü elemeit az ablakba (játékos neve, „new game” és „scoreboard” gombok) a draw_piece és write_text függvényeket is meghívva.

A *draw_difficulty_select* függvény draw_menu-höz hasonlóan a nehézségválasztáshoz szükséges „easy”, „normal” és „hard” gombokat rajzolja ki.

Az *input_text* függvény hívásával egy általa megjelenített szövegdobozba írhatunk, amit aztán be is ír a paraméterként kapott sztringbe az enter lenyomása után. (InfoC oldalról)

A *cut_spaces* függvény kivágja a paramétereként kapott sztringből a szóközöket.

states.c

Az *st_menu* függvény többek közt a korábbi függvények hívásával kirajzolja a menüt, valamint lehetővé teszi, hogy a felhasználó a bal egérgomb kattintásával elérje a megjelenített menüpontokat, vagy átnevezhesse magát. Ehhez az SDL esemény alapú vezérlését használja. A state változó értékének megváltoztatásával éri el, hogy a main eseményhurkában a megfelelő állapotba ugorjon a program.

Az *st_dificulty_select* függvény az st_menu-höz hasonlóan megjeleníti a nehézségválasztás menüpontot, és figyeli, hogy a játékos ráklikkelt-e valamelyik gombra. A state változó értékének megváltoztatásával éri el, hogy a main eseményhurkában a megfelelő állapotba ugorjon a program.

Az *st_game* függvény a fentiekhez hasonlóan megjeleníti a megfelelő játéknézetet. Létrehozza a kiválasztott nehézségi szintnek megfelelő számú űrlényt dinamikusan foglalt memóriaterületen, ha még ez nem történt meg, illetve a balra és jobbra nyilak, valamint a space lenyomását és felengedését figyelve mozgatja a játékos űrhajóját és lő vele a Player típusú struktúra megfelelő bool változói értékének megváltoztatásával. A game struktúra state változója alapján beállítja a nem állandó sebességeket (a játékos lövedéke állandó sebességű). Mozgatja az űrlényeket jobbra-balra a pálya két széle között, az űrlények lövedékeit mozgatja. Ha teljesül a játék végét jelentő valamelyik feltétel, akkor a statet átállítja, hogy továbblépjen a program a játék vége nézetbe.

A *swarm_move* függvény felel az űrlények jobbra-balra mozgatásáért.

A *shoot_p_laser* függvény vizsgálja, hogy a játékos lőtt-e, és ha igen, megjeleníti és mozgatja a játékos lövedékét, valamint figyeli, hogy a lövedék kiment-e a pályáról, akkor hamisra állítja a játékos lövedékének kilőtt állapotát, ugyanezt teszi, ha űrlényt talált a lövedék, ebben az esetben viszont hozzáad a játékos pontjaihoz 10-et, és az eltalált űrlény életállapotát hamisra változtatja.

A *shoot_a_laserX* függvény vizsgálja, hogy az űrlények 1-es vagy 2-es lövedékét ki kell-e lőni. A shoot_p_laser függvényhez hasonló, csak számolnia kell, hogy lőhetnek-e adott időben az űrlények, illetve generál egy véletlenszámot, hogy melyik űrlény lőjön.

Az *a_laserX* függvény mozgatja az űrlények lövedékeit, és ha azok kimentek a pályáról, akkor hamisra állítja a lövedékek kilőtt állapotát jelző bool változót. Ha eltalálták a játékos űrhajóját, akkor az űrlények győzelmét jelző bool változót állítja igazra, illetve a statet átállítja, hogy a játék vége nézet következzen.

Az *st_end_of_game* függvény felszabadítja a maradó űrlények struktúráit, ha az űrlények nyertek, alaphelyzetbe állítja az új játék indításához szükséges változókat, illetve beolvassa a scores.txt sorait egy-egy sztringbe, melyek egy tömbben vannak. Kirajzolja a játék vége nézetet, kiszámolja az elért pontszámot, és ha ez megvan, akkor a végleges pontszám meglétét jelző bool változót igazra állítja, mert nem kell újra kiszámolni a pontszámot, és ki is írja a pontszámot, hogy a játékos is láthassa. Ezután a sztringekből kikeresi a pontszámokat tartalmazó részt, (ez a szóköz után van,) dinamikusan foglal memóriaterületet 5 int változónak, majd azokba beírja a sztringekből a pontszámokat egy átalakítás után (atoi()). Ezt követőn összehasonlítja a végleges pontszámot mindegyikkel, és ha szükséges beírni a legjobb 5 közé, akkor beírja a scores.txt megfelelő sorába, a felette lévőket visszaírja ugyanoda, ahol voltak, az alatta lévőket lejjebb tolja egy sorral, és a korábbi 5. legjobbat törli, így továbbra is csak 5 sor marad a fileban. Ezután igazra állítja az összehasonlítás kész állapotát jelző bool változót, és

bezárja a filet. A megjelenített „menu” gombra kattintás esetén átállítja a statet, hogy visszalépjen a menübe.

Az *st_scoreboard* függvény megjeleníti a dicsőséglistát, melyhez beolvassa a scores.txt-ből a legjobb 5 pontszámot elért játékos nevét és pontszámát, és azokat kiírja a képernyőre. A megjelenített „menu” gombra kattintás esetén átállítja a statet, hogy visszalépjen a menübe.