



AirCombat

Fejlesztői dokumentáció

1. Program felépítése

A program a GLUT keretrendszert használja a platformfüggetlen megjelenítéshez. Ennek szabványos event függvényeit implementálja a szükséges műveletekhez. A World a fő objektum, minden más ebből kapcsolódik hierarchikusan.

onDisplay()

világ felrajzolása a képernyőre

onIdle()

világ animálása, onDisplay() hívása

onReshape()

ablakméretezésnél a kép igazítása

keyboard()

billentyűparancsok kezelése

2. World

A világot reprezentáló osztály. Fontosabb függvényei:

Init()

létrehozza az objektumokat, betölti a modelleket, valamint multiplayer esetén a socketet

LoadTextures()

betölti a textúrákat

Build()

felépíti a képszíntér megvilágítását, betölti a földfelszínt, beállítja a ködöt

Animate()

lépteti a modelleket a Move() függvényeik segítségével

Draw()

kirajzolja a színteret

drawGround()

kirajzolja a földfelszínt

drawSky()

kirajzolja az eget

drawWater()

kirajzolja a vizet

drawSun()

kirajzolja a napot

drawMiniMap()

kirajzolja a térképet

FireRocket(AirCraft Aircraft)*

kilő egy rakétát a megadott repülővel

CreateAirCraft()

létrehoz egy új repülőt

ExecRemoteCommand()

végrehajt egy a szervertől kapott parancsot

ExecCommand()

végrehajt egy helyi, vagy távoli parancsot

SendCommand()

parancsot küld a szervernek

QueryServer()

lekérdezi a szervert új parancsokat várva

3. AirCraft

A repülőt reprezentáló osztály. Fontosabb függvényei:

Move()

kiszámítja a következő pozíció koordinátáit

Draw()

kirajzolja Display Listtel a modellt

FireRocket()
kilő egy rakétát

4. Rocket

A rakétát reprezentáló osztály. Fontosabb függvényei:

Move()
kiszámítja a következő pozíció koordinátáit

Draw()
kirajozlja Display Listtel a modellt

5. Bitmap

A képet reprezentáló osztály. Fontosabb függvényei:

Save()
elmenti egy char tömböt 8 bites képet bmp formátumban

Load()
betölt egy 24 bites bitképet

6. BoundingBox

A határoló téglatestet reprezentáló osztály. Fontosabb függvényei:

bool Intersect(BoundingBox other)*
true ha a másik BB-vel van metszet, amúgy false

7. Config

A beállításokat reprezentáló osztály. Fontosabb függvényei:

Load()
betölti az AirCombat.cfg fájlból az alábbi formátumú beállításokat:
címké=érték

GetStr(char key)*
a key címkéjű beállítás értékét adja vissza

vector<char> Tokenize(char* str, char* delim)*
feldarabolja az str szöveget a delim határoló mentén

8. Model

A modelt reprezentáló osztály. Fontosabb függvényei:

Draw()

kirajzolja a modelt a Display List alapján

MakeDisplayList()

Display Listet generál, és abban rajzolja a modelt

drawBurn()

tűzcsóvát rajzol a model mögött

drawExplosion()

robbanást rajzol

9. Server

A program támogatja a multiplayer játékmódot. Ilyenkor az egyik gépen az `./AirCombat server` paranccsal futtatandó. A kapcsoló gépeket pedig az `./AirCombat client` paranccsal lehet bekapcsolódni a játékba.

A szerver először létrehozza a nem blokkoló socket a `socket/bind/listen` parancsokkal. Majd egy végtelenített ciklust futtat. Amiben ha az `accept` visszatér, akkor létrehoz egy szálat a kliens részére.

Ez az `Answer()` függvény. Felvéve egy új queue-t a kliensnek szóló parancsokhoz, ez a szál is végtelen ciklusba kerül. Ebben ha a `ReceiveLine()` visszatér, akkor parancs érkezett a klientsől. Ha ez nem üres, akkor ezt továbbítja a többi klient felé. Majd ha van az aktuális kliens üzenetsorába parancs, akkor azt továbbítja felé a szerver.

Az üzenetek felépítése:

COMMAND|ID

ahol a COMMAND az üzenet típusa, az ID pedig a kliens azonosítója

10. Kliens

Multiplayer működés esetén a World osztály `QueryServer()` metódusa felelős a hálózati kommunikációért. Időnként lekérdezi a szervert új parancsokat várva. Ha van ilyen, akkor azt az `ExecRemoteCommand()` értelmezi, és hajtja végre.

Az `ExecRemoteCommand()` az `ExecCommand()` metódust használja fel, ami normál esetben is felelős a felhasználói felületről érkező parancsok végrehajtásáért.

Minden kliens minden felhasználói felületről érkező parancsa(ami befolyásolja a modellek helyzetét) továbbításra kerül a szerver felé. A szervert ezeket szétküldi a többi kliens felé, azok pedig végrehajtják a megfelelő modellen.

Új játékos érkezésekor az küld egy STATUS üzenetet. Eerre mindegyik kliens az aktuális pozíciójával válaszol. Ennek alapján az új játékos képes felépíteni a világot.