Space Shooting Game

HiScore : 03500

Start

manual

ResetHiScore

矢印キーで選択。spaceキーで決定。

概要

基本情報

タイトル　　：Space Shooting Game
ジャンル　　：シューティング
制作期間　　：2カ月
制作人数　　：1人
使用言語　　：C言語

使用ツール

VisualStudio2022,DxLib

対応プラットフォーム

Windows

制作意図

1年次の作品展に展示するため。
当たり判定やDxLibについて学ぶため。
特に当たり判定は理解のしやすさに
注力しました。

ゲーム内容

60秒間、上から飛んでくる敵を
弾を打って倒すシューティングゲーム。
敵に触れたらゲームオーバー。

アピールポイント

1.当たり判定

2.メニュー操作

3.ハイスコア

4.エフェクト

5.背景スクロール

1．当たり判定

プレイヤーもしくは弾(円形)と敵(四角形)の当たり判定は
以下の方法で実装しました。

①敵の左右の端のx座標 ＋ プレイヤーの半径を求める。
②敵の上下の端のy座標+ プレイヤーの半径を求める。
敵の大きさにプレイヤーの大きさを足した値が求まります。
③プレイヤーの中心x, y座標がその範囲に入っているか比較。
④衝突している場合は1、衝突していない場合は0を返す。

該当スクリプト:
プレイヤーと敵
Char.cpp　177行目 ～ 197行目

該当スクリプト:
プレイヤーの弾と敵
Shot.cpp　49行目 ～ 68行目

```
177     // プレイヤーとの衝突判定をとる関数(衝突していたら1を返す)
178     int Enemy::Col(int xx, int yy, int ww, int hh)
179     {
180         if (live == OFF)
181             return 0;
182         // プレイヤー(円)と敵(四角形)の衝突判定
183         int r = (ww + hh) / 4;  // 円の半径
184         int x1 = (x - w/ 2) - r;    // 左端x座標
185         int y1 = (y - h / 2) - r;   // 上端y座標
186         int x2 = (x + w / 2) + r;   // 右端x座標
187         int y2 = (y + h / 2) + r;   // 下端y座標
188
189         // 四角形に半径を追加したエリア内に円の座標があったら衝突している
190         if (xx > x1 && xx < x2 && yy > y1 && yy < y2)
191         {
192
193             return 1;
194         }
195
196         return 0;
197     }
```

```
48
49      // 敵との衝突判定をとる関数(衝突していたら1を返す)
50      int Shot::Col(int xx, int yy, int ww, int hh)
51      {
52          if (state == OFF)
53              return 0;
54
55          int x1 = (xx - ww / 2) - r; // 左端x座標
56          int y1 = (yy - hh / 2) - r; // 上端y座標
57          int x2 = (xx + ww / 2) + r; // 右端x座標
58          int y2 = (yy + hh / 2) + r; // 下端y座標
59
60          // 四角形に半径を追加したエリア内に円の座標があったら衝突している
61          if (x > x1 && x < x2 && y > y1 && y < y2)
62          {
63              state = OFF;
64              return 1;
65          }
66
67          return 0;
68      }
69
```

２．メニュー操作

メニュー操作ではカーソルが真ん中をスルーすることを防ぐために以下の処理をしました。

①現在時間から最終入力時間の格納された変数(start)を引いて、最後の入力から経過した時間を求める。
②その経過時間が80よりも大きいなら、入力処理をする。そうでないなら、入力処理をしない。

該当スクリプト:
main.cpp　108行目 ～ 151行目

```
108     if (GetNowCount() - start > 80)
109     {
110         if (CheckHitKey(KEY_INPUT_W) || CheckHitKey(KEY_INPUT_UP)) Key += 1;          // 上キーが押されたらカーソルを上げる
111         if (CheckHitKey(KEY_INPUT_S) || CheckHitKey(KEY_INPUT_DOWN)) Key -= 1;      // 下キーが押されたらカーソルを下げる
112         if (CheckHitKey(KEY_INPUT_SPACE)) Space = 1;    // スペースキーが押されたら決定(1)にする
113
114         if (Key >= 2)
115         {
116             CursorX = 280;
117             CursorY = 237;
118             if (Space == 1)
119             {
120                 State = GAME;
121                 WaitTimer(500);
122                 break;  // ループを抜けてゲームをスタートする
```

```
137         if (Key <= 0)
138         {
139             CursorX = 280;
140             CursorY = 320;
141             if (Space == 1)
142             {
143                 pl.ResetHiScore();
144                 DrawString(100, 400, "ハイスコアをリセットしました。", StrColor);
145                 ScreenFlip();
146                 Space = 0;
147                 WaitTimer(1000);
148             }
149             Key = 0;
150         }
151         start = GetNowCount();
152     }
```

3.ハイスコア

ハイスコアをテキストファイルに書き込むことで、
ゲームを閉じてもハイスコアが保存されるようにしました。

ハイスコアの設定は以下の方法で実装しました。
①ファイルポインタにテキストファイルのアドレスを格納。
②中に書き込みがあった場合は中身を取得して変数に格納。
中がNULLもしくはハイスコアが更新された場合は、
中身を書き換える。

該当スクリプト:
ハイスコアのリセット
Char.cpp　115行目 ～ 121行目

該当スクリプト:
ハイスコアの設定
Char.cpp　123行目 ～ 139行目

該当スクリプト:
ハイスコアの取得
Char.cpp　141行目 ～ 152行目

```
115  void Player::ResetHiScore(void)
116  {
117      FILE* fp;
118      fp = fopen("score.txt", "w");
119      fprintf(fp, "%d", 0);
120      fclose(fp);
121  }
122
123  void Player::SetHiScore(void)
124  {
125      FILE* fp;
126      int hiScore;
127      fp = fopen("score.txt", "r");
128      if (fp != NULL) // ファイルの中に書き込みがあった場合
129      {
130          fscanf(fp, "%d", &hiScore);
131          fclose(fp);
132      }
133      if (fp == NULL || score > hiScore)  // ファイルの中が空もしくはハイスコアが更新された場合
134      {
135          fp = fopen("score.txt", "w");
136          fprintf(fp, "%d", score);
137          fclose(fp);
138      }
139  }
140
141  int Player::GetHiScore(void)
142  {
143      FILE* fp;
144      int hiScore;
145      fp = fopen("score.txt", "r");
146      if (fp == NULL)
147          return 0;
148      fscanf(fp, "%d", &hiScore);
149      fclose(fp);
150
151      return hiScore;
152  }
```

４．エフェクト

エフェクトのアニメーションは2種類の円を使って
以下の方法で実装しました。

①経過時間が一定時間を超過したかどうか比較する。
②超過していたら、それぞれの円の半径を大きくする。
③半径が最大値を超えたら0にして、周回数をカウント。
④周回数が3を超えたら、アニメーションを終える。

該当スクリプト:
Effect.cpp

該当スクリプト:
エフェクトのアニメーション
Effect.cpp　42行目 ～ 72行目

```
42      // アニメーション関数
43      void Effect::Animation(void)
44      {
45          // 経過時間がSPANに満たなければ実行しない
46          if (GetNowCount() - start < SPAN)
47              return;
48
49          // 半径を大きくする
50          bigR += BIGR_SPEED;
51          smallR += SMALLR_SPEED;
52
53          // 半径が最大半径を越えたら半径を0にする
54          if (bigR > MAX_R)
55          {
56              bigR = 0;
57              count++;    // 周回数を増やす
58          }
59          // 半径が最大半径を越えたら半径を0にする
60          if (smallR > MAX_R)
61              smallR = 0;
62
63          // 周回数が3以上になったらアニメーションを終える
64          if (count >= 3)
65          {
66              flag = OFF;
67              bigR = 0;
68              smallR = 0;
69              count = 0;
70          }
71          start = GetNowCount();
72      }
```

５．背景スクロール

背景スクロールは以下の方法で実装しました。

①背景のy座標にスクロールスピードを足して
表示位置を下に下げる。
②y座標が0になったら端まで描画したことになるので
y座標から背景のサイズ分の値を引いて上に戻す。

該当スクリプト:
backGround.cpp

該当スクリプト:
背景のスクロール
backGround.cpp

```
22      // スクロール関数
23      void backGround::Scroll(void)
24      {
25          y += BG_SPEED;  // 背景を下に移動させる
26
27          if (y >= 0) // 背景が消えたら
28              y -= BG_SIZE;   // 画像サイズ分上に戻す
29      }
```

反省点

反省点

①コメントが少ないため可読性が低いこと。

②重複処理が多いため拡張性が低いこと。

③main.cppの行数が多いため、
バグ対応が難しいこと。

④締め切りの直前まで制作をしていたため、
処理が煩雑になっている部分があること。

まとめ:
可読性と拡張性に欠けるプログラムになったこと。

改善点・学び

①こまめにコメントを書くことで可読性を上げること。

②重複処理は関数化する工夫を行うこと。

③仕様を決めてから制作に取り掛かることで
main.cppの行数を減らすこと。

④期限に余裕をもってスケジュールを組むこと。

まとめ:
制作前に仕様を決めることで、整理されたコードを
書けるようにしていきたい。