```
#include <string>
#include <vector>
#include <fstream>
#include "DxLib.h"
#include "Utility.h"
#include "GameScene.h"
#include "Stage.h"
Stage::Stage(GameScene* scene)
   mGameScene = scene;
void Stage∷Init(int stageNo)
   LoadDivGraph(
       "Image/StageChip.png",
       4,
       2, 2,
       BLOCK_SIZE, BLOCK_SIZE,
       &mImages[0], true);
   // オブジェクト無し(芝生)
   mImageOutFloor = mImages[0];
   // 床
   mImageInFloor = mImages[1];
   // 内壁
   mImageInWall = mImages[2];
   // 外壁
   mImageOutWall = mImages[3];
   // コードでステージを設定する
   //SetStage(stageNo);
   // 外部データを使用して、ステージを設定する
   LoadData(stageNo);
}
void Stage::Update(void)
{
}
void Stage∷Draw(void)
```

{

```
// ゲームエリアの描画
int mapNo;
int tmpImage = -1;
for (int y = 0; y < MAP_SIZE_Y; y++)
{
    for (int x = 0; x < MAP_SIZE_X; x++)
       mapNo = mMap[y][x];
        switch (mapNo)
        {
       case -1:
           // オブジェクト無し(芝生)
           tmpImage = mImageOutFloor;
           break;
       case 1:
           // 床
           tmpImage = mImageInFloor;
           break;
       case 2:
           // 内壁
           tmpImage = mImageInWall;
           break;
       default:
           break;
       }
       DrawGraph(
           GAME\_AREA\_X + (x * BLOCK\_SIZE),
           GAME\_AREA\_Y + (y * BLOCK\_SIZE),
           tmpImage, true);
   }
}
// ゲームエリア外側の描画
int scSizeX = SCREEN_SIZE_X / BLOCK_SIZE;
int scSizeY = SCREEN_SIZE_Y / BLOCK_SIZE;
// 横
for (int x = 0; x < scSizeX; x++)
{
    DrawGraph(
       x * BLOCK_SIZE, 0 * BLOCK_SIZE,
       mImageOutWall, true);
```

```
DrawGraph(
        x * BLOCK_SIZE, 1 * BLOCK_SIZE,
        mImageOutWall, true);
     DrawGraph(
        x * BLOCK_SIZE, (scSizeY - 1) * BLOCK_SIZE,
        mImageOutWall, true);
  }
  // 縦
  for (int y = 0; y < scSizeY; y++)
   {
     DrawGraph (
        0 * BLOCK_SIZE, y * BLOCK_SIZE,
        mImageOutWall, true);
     DrawGraph(
         (scSizeX - 1) * BLOCK_SIZE, y * BLOCK_SIZE,
        mImageOutWall, true);
  }
}
void Stage::Release(void)
{
  DeleteGraph(mImageInWall);
  DeleteGraph(mImageInFloor);
  DeleteGraph(mImageOutWall);
  DeleteGraph(mImageOutFloor);
}
bool Stage∷IsCollosion(Vector2 mapPos)
  return mMap[mapPos. y] [mapPos. x] > 1;
}
void Stage∷SetStage(int stageNo)
{
  switch (stageNo)
  case 2:
   {
     int tmpMap[MAP_SIZE_Y][MAP_SIZE_X] = {
                                { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                                0, 0, 0, 0, 0, 0, 0, 0, 0,
                                                       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
                                0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
```

```
{ 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, }
          { 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    9, 9, 1, 1, 1, 9, 9, 9, 9, 9,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         { 0, 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    1, 1, 1, 1, 1, 1, 1, 1, 9,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
          1, 1, 1, 1, 1, 1, 1, 1, 9,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
          1, 1, 1, 9, 9, 1, 1, 1, 1,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         { 0, 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    1, 1, 1, 9, 9, 9, 1, 1, 1, 1,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         { 0, 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    1, 1, 1, 9, 1, 1, 1, 1, 1, 1,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         { 0, 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    1, 1, 1, 9, 1, 1, 1, 1, 1,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, }
          { 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    1, 1, 1, 9, 1, 1, 1, 1, 1, 1,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         { 0, 0, 0, 0, 0, 0, 0, 0, 9,
                                    9, 9, 9, 9, 9, 9, 9, 9, 9,
                                                             9, 0, 0, 0, 0, 0, 0, 0, 0, 0, },
         };
      for (int y = 0; y < MAP_SIZE_Y; y++)
      {
         for (int x = 0; x < MAP_SIZE_X; x++)
            mMap[y][x] = tmpMap[y][x];
         }
      }
   }
}
void Stage∷LoadData(int stageNo)
{
   // ファイルパスを取得する
   std::string filePath =
      mGameScene->GetCsvPathGround(stageNo);
   // ファイルを読み込む
   std::ifstream ifs(filePath);
   // 1行ずつ読み込む
   int y = 0;
   std∷string line;
   while (getline(ifs, line))
   {
      // Yで分割されたline
```