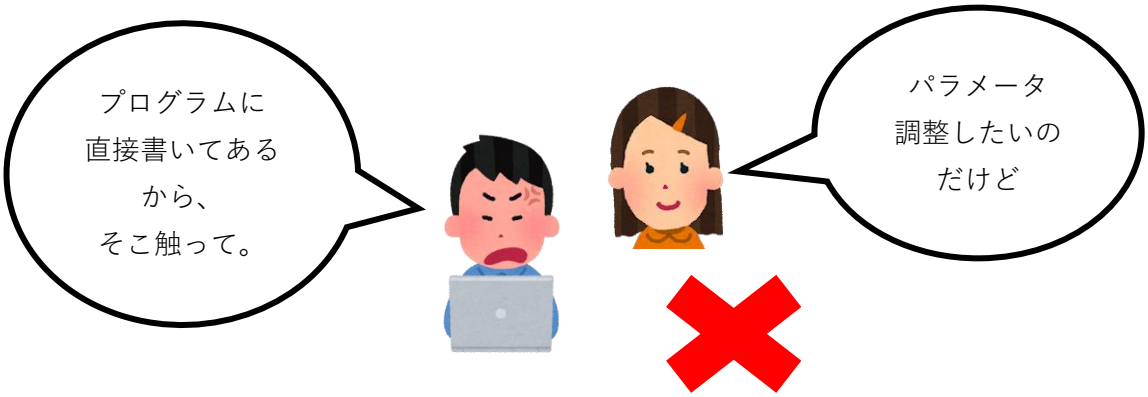
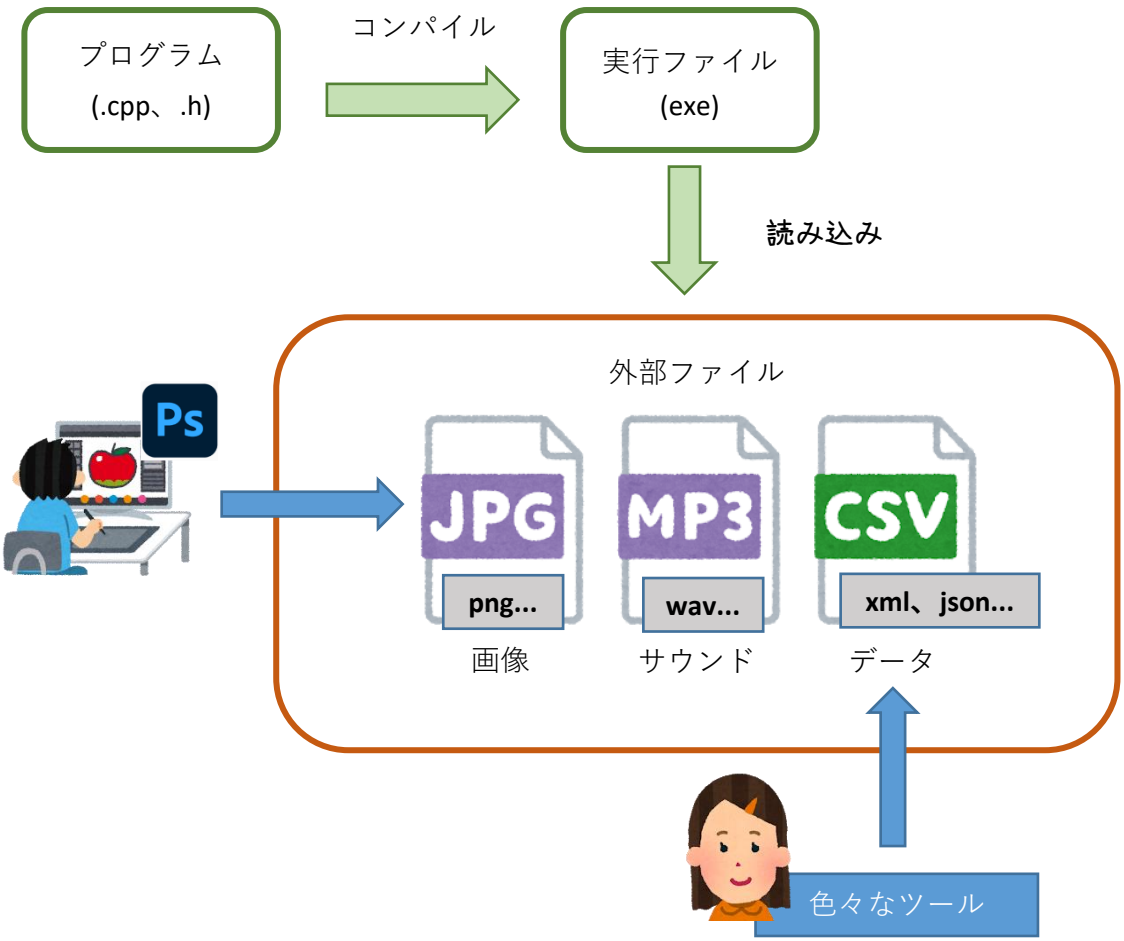
外部ファイルの読み込み

ステージ情報や、キャラクターのパラメータ、敵の経験値などなど。。。
プログラムに直接記述すると、メンテナンスが大変になります。

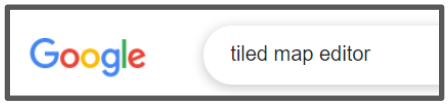
実際のゲーム開発の現場では、
そのあたりはプランナーがバランス調整を行いますので、
プランナーが操作できる形する必要があります。



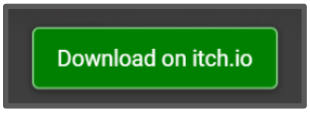


今回は、「Tiled Map Editor」を使用して、ステージデータを作成していき、
それをプログラムから読み込んで、ゲームステージとして、組み上げていきます。

「Tiled Map Editor」
https://www.mapeditor.org/



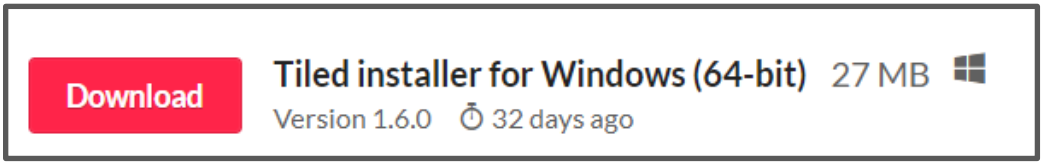
① サイトにいって、ダウンロード



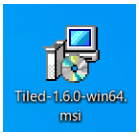
② 表示される別ページにて、ダウンロード

③ サポートは必要ありません！

④ Windows64bit版をダウンロード

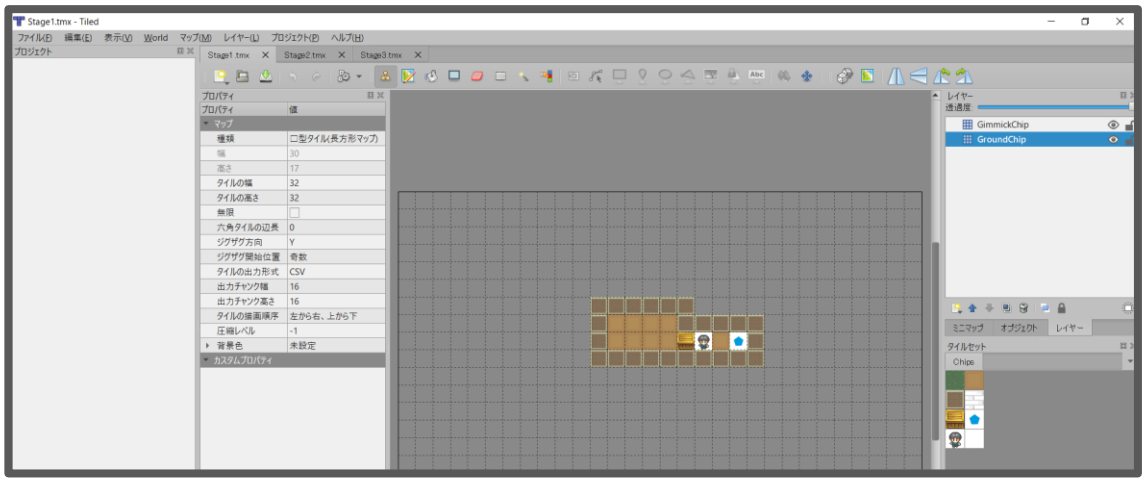


⑤ インストーラをダブルクリックして、インストール

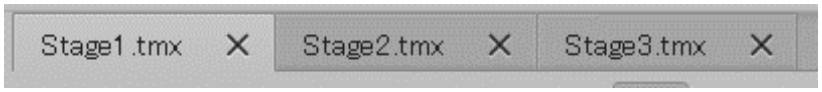


早速使ってみましょう！

プロジェクト配下の[StageData]->[AsoBaseマップ作成]
Stage1.tmx　をダブルクリック



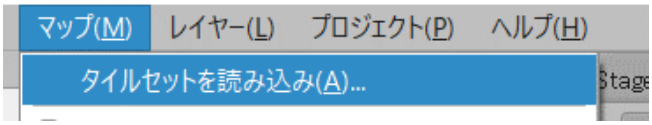
画面が起動します。
ついでに、Stage2.tmx、Stage3.tmxもダブルクリックして
起動しておきます。

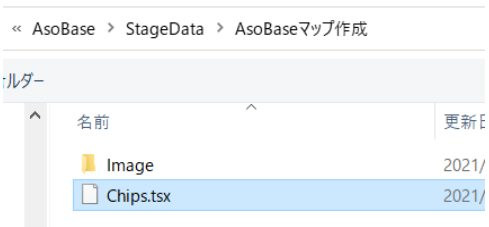


3ステージ分のデータが読み込まれました。

習うより慣れろ、Stage4を一緒に作っていきましょう。

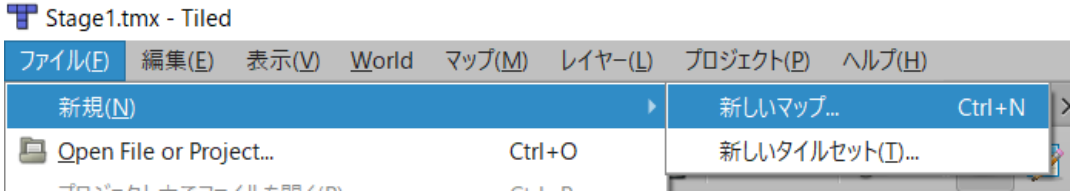
タイルセットが表示されていない方は、





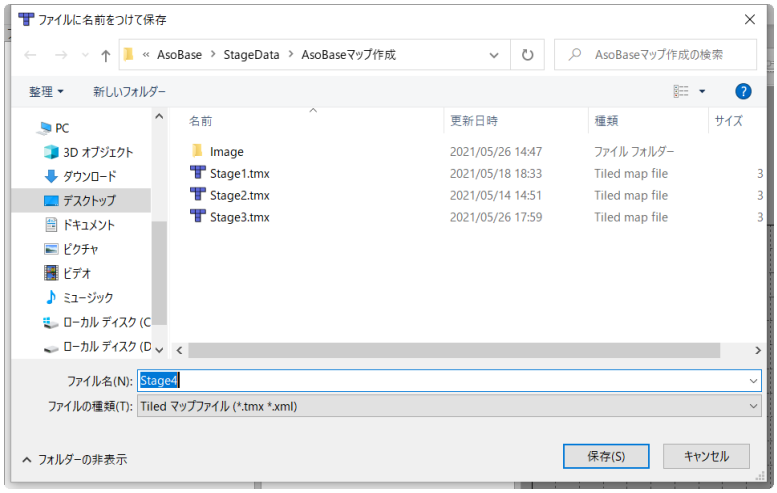
Chips.tsxを
読み込んでください。

① 新しいマップの作製



② ゲームのマップに合わせて、マップサイズを設定する

③ ファイルの名前はStage4



場所は、

プロジェクト配下の[StageData]->[AsoBaseマップ作成]

④ レイヤーを用意する

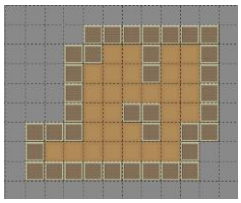
AsoBaseでは、２つのレイヤー(層)を使用していきます。

１つは、「GroundChip」

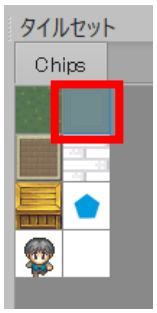
レイヤー名を変更してください。

このレイヤーでは、床と内壁を設置します。

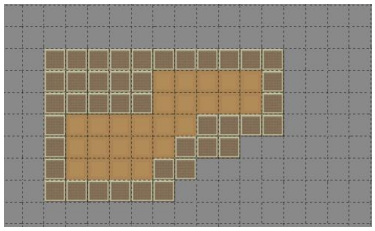
Stage3ではこんな感じ。



２つ目のレイヤーを作る前に、
自分なりにステージを作成してみましょう。



右下のタイルセットから床や内壁を選択して、
メイン画面をポチポチ押していくだけです。



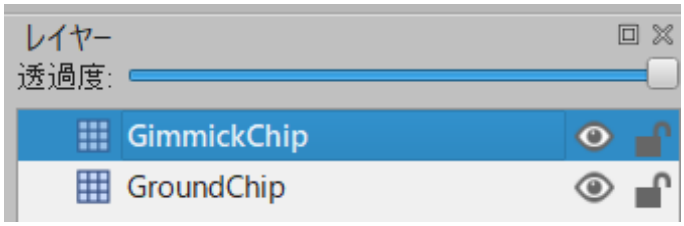
削除したい場合は、

消しゴムツールを使います。

⑤　２つ目のレイヤーを用意する

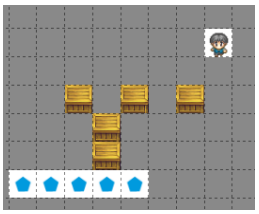
新しいレイヤーを作成します。

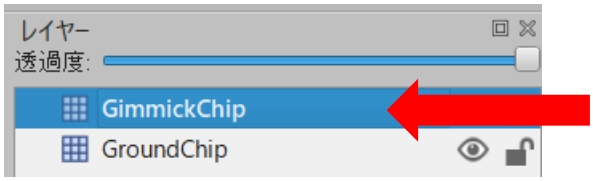
名前を、「GimmickChip」にします。



このレイヤーでは、
プレイヤー、荷物と、荷物の格納場所を設置します。

Stage3ではこんな感じ。



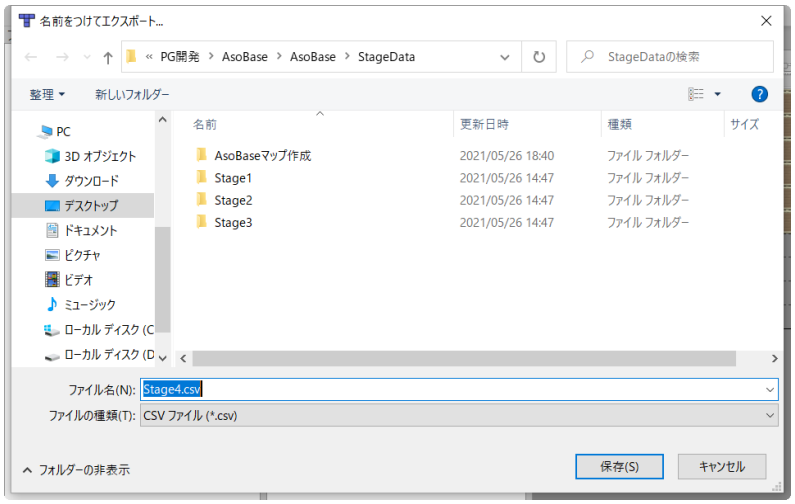


必ず、設置したいレイヤーが
選択されているか、
事前に確認してください。

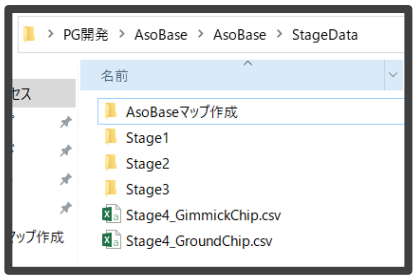
これで、ステージ作りは完了です。
こういったツールだったら、プログラマー以外の職種の人でも、
操作可能ですね。

それでは、いよいよ、データファイルを出力していきます。
名前をつけてエクスポート

プロジェクト配下の[StageData]フォルダを指定して、
保存ボタンを押します。



プロジェクト配下の[StageData]フォルダの中を見てみると、

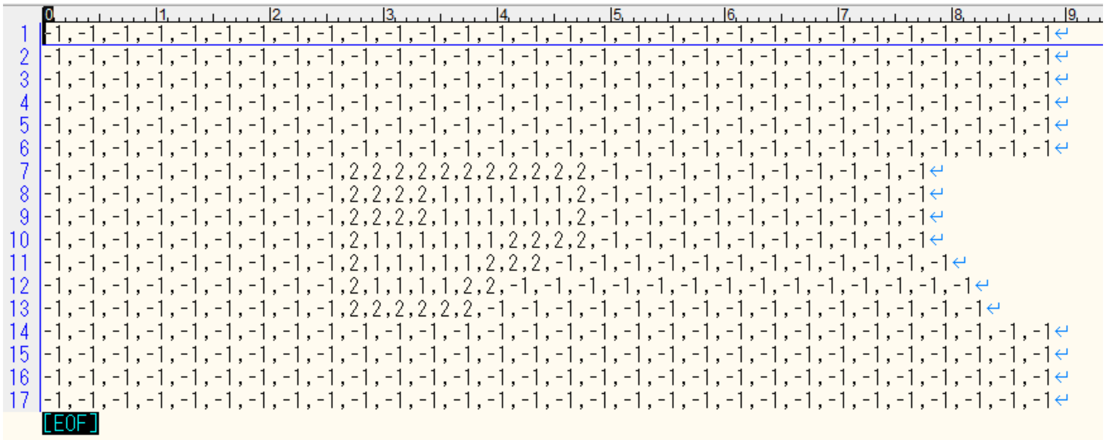


２つCSVファイルができました。
作成したレイヤーごとに、データが出力されます。

CSVデータとは？

カンマ(,)区切りに分けられたデータのことです。

先ほど出力したCSVファイルの中を見てみると、



```
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,2,2,2,2,2,2,2,2,2,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,2,2,2,1,1,1,1,1,1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,2,2,2,1,1,1,1,1,1,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,1,1,1,1,1,1,2,2,2,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,1,1,1,1,1,2,2,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,1,1,1,1,2,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,2,2,2,2,2,2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
[EOF]
```

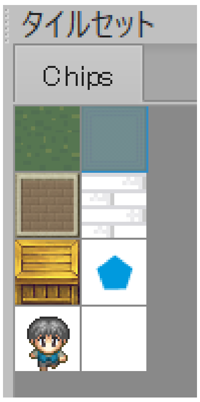
このようなデータになっています。

マップの広さを横30、縦17に設定しましたので、
1行あたり、30個のデータが、17行出来上がります。

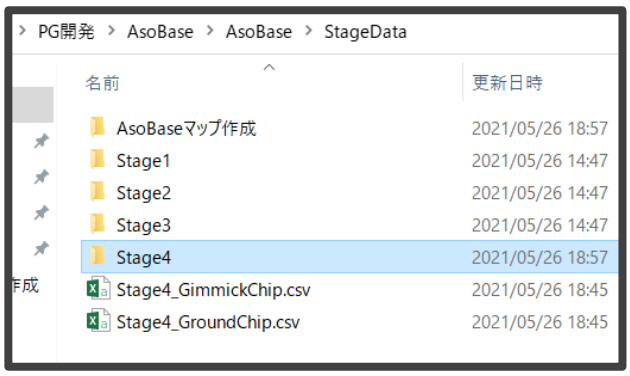
↑ は、「GroundChip」ですが、
-1が未設定
1が床
2が内壁という意味です。

「GimmickChip」の方は、
-1が未設定
4が荷物
5が荷物置き場
6がプレイヤーになります。

事前に用意した、
タイルセットの順番です



仕上げに、



「Stage4」のフォルダを作成して、
先ほどの２つのCSVを作成したフォルダの中に移動させます。

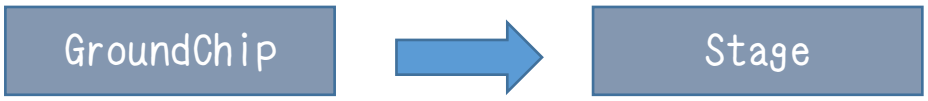
移動させたら、それぞれ、名前を変更してください。

GroundChip.csv

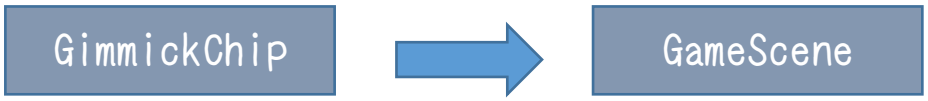
GimmickChip.csv

それでは、いよいよ、この外部ファイルを読み込む、
プログラムを作成していきましょう。

まずは準備を。



床、内壁　　床、内壁



プレイヤー、
荷物、荷物置き場　　基本、全部

各外部データを上記のクラスで、それぞれ読み込んでいきます。
読み込む際の、ファイルパスやファイル名を間違えないように、
関数を作っておきましょう。

GameScene

```
// ステージ構成ファイルパス
std::string GetCsvPathGround(int stageNo);
// ギミック構成ファイルパス
std::string GetCsvPathGimmick(int stageNo);
```

次に、Stageの読み込み処理を作成していきますが、
カンマ区切りのデータを動的配列に変換する便利な処理のご紹介。

```
std::vector<std::string> Utility::Split(std::string& line, char delimiter)
{
    std::istringstream stream(line);        → 文字操作が便利に行える
    std::string field;
    std::vector<std::string> result;

    while (getline(stream, field, delimiter)) {
        result.push_back(field);
    }                                       getline
    return result;                              ストリームから改行文字が現れるまでか、
}                                               仮引数delimで指定された文字までの
                                                文字列を入力する。
```

いよいよ、読み込み処理。

```
void Stage::LoadData(int stageNo)
{
   std::string filename =
      mGameScene->GetCsvPathGround(stageNo);

   std::ifstream ifs(filename);

   int y = 0;
   std::string line;
   while (getline(ifs, line)) {

      std::vector<std::string> strvec = Utility::Split(line, ',');

      for (int x = 0; x < strvec.size(); x++) {
         mMap[y][x] = stoi(strvec.at(x));
      }

      y++;                              stoi… str to int
                                              文字型を数字型にしてくれます。
   }
}
```

これで、CSVファイルから、データを読み込んで、
ステージ描画用の２次元配列に値を格納することができました。