

# GameManager マップ描画リファクタリング

実装要件ドキュメント v1.0 2026/02/19

## 1. 背景と目的

現在の GameManager.cs では、マップ表示の UpdateMapDisplay() において全マスを一重ループして毎回プレイヤー位置をチェックしている(力押し方式)。今後 NPC やエネミーの配置も予定しているため、拡張性を考慮した設計へリファクタリングする。

## 2. 現状の問題点

- 全マスを一重ループして毎回プレイヤー位置をチェックしている(非効率)
- mapData(string[]) はイミュータブルなためピンポイント書き換えができない
- NPC・エネミーなど複数キャラクターの描画に対応しにくい構造になっている

## 3. 設計方針

「静的な元データ」と「動的な盤面データ」を分離する考え方を採用する。

- baseMapData (static readonly string[]) : マップの設計図。絶対に変更しない。
- gridData (char[][]): 実際の盤面。プレイヤー・NPC・エネミーなどが乗る動的なデータ。

描画タイミングで baseMapData から gridData をリセットし、その上にキャラクターをレイヤー順に重ね書きするアーキテクチャとする。

## 4. 実装要件

### 4-1. baseMapData の定義

変更不可の元マップデータを static readonly で定義する。

```
private static readonly string[] baseMapData = {  
    "#####  ###",  
    "#...####.#",  
    // ...  
};
```

### 4-2. gridData の初期化

Start() または InitGrid() で baseMapData を char[][] に変換して初期化する。

```
private char[][] gridData;

void InitGrid()
{
    gridData = new char[baseMapData.Length][];
    for (int y = 0; y < baseMapData.Length; y++)
        gridData[y] = baseMapData[y].ToCharArray();
}
```

### 4-3. ResetGrid()

描画のたびに baseMapData から gridData を再構築する。これにより「前のキャラクターを消す」処理が不要になり、上書きミスを防ぐ。

```
void ResetGrid()
{
    for (int y = 0; y < baseMapData.Length; y++)
        gridData[y] = baseMapData[y].ToCharArray();
}
```

### 4-4. PlaceCharacters()

キャラクターをレイヤー順(NPC → エネミー → プレイヤー)に gridData へ配置する。プレイヤーは最後にかくことで常に最前面に表示される。

```
void PlaceCharacters()
{
    foreach (var npc in npcList)
        gridData[npc.pos.y][npc.pos.x] = npc.symbol;

    foreach (var enemy in enemyList)
        gridData[enemy.pos.y][enemy.pos.x] = enemy.symbol;

    // プレイヤーは最後（最優先）
    gridData[playerPosition.y][playerPosition.x] = '@';
}
```

### 4-5. UpdateMapDisplay() の更新

描画関数を ResetGrid → PlaceCharacters → RenderGrid の3ステップに整理する。

```
void UpdateMapDisplay()
{
    ResetGrid();           // 盤面をbaseMapDataでリセット
    PlaceCharacters();
    RenderGrid();
}
```

```
PlaceCharacters();    // キャラクターを重ね書き  
RenderGrid();        // TMP_Textに出力  
}
```

## 5. 拡張ポイント

この設計はキャラクター追加に対して非常に柔軟に対応できる。PlaceCharacters() にリストを追加するだけで新しい要素に対応できる。

- NPC追加: npcList にオブジェクトを追加するだけ
- エネミー追加: enemyList に追加するだけ
- アイテム配置: itemList を追加してレイヤーに組み込む
- 描画優先度の変更: PlaceCharacters() 内の順序を変えるだけ

## 6. 対象ファイル

- Assets/GameManager.cs

---

※ 本ドキュメントはスノハラ×クーちゃんの壁打ちセッションをもとに作成