



BANDAI NAMCO Studios

ゲームプログラマー 1day インターンシップ 2019冬

戦車対戦ゲームの
AIを作ろう！

株式会社バンダイナムコスタジオ
BANDAI NAMCO Studios Inc.

◀ 戦車ゲームのAIを作ろう！



BANDAI NAMCO Studios

Unity公式チュートリアル『TANKS!』を使って、対戦ゲームを作ります。
ただし、皆さんが今回実装するのは **戦車のAI** です。

AI対戦を行う土壌はすでに完成しているので、頑張ってAIを実装してください！

と、さっそくAI実装に入ってもらいたいところですが、AI対戦が出来るように改修した過程で発生した **プロのテクニック** を皆さんにご紹介します。

「プロのテクニック（笑）」が今後の皆さんの創作活動に役立てば幸いです。

Unity公式チュートリアル『TANKS!』



BANDAI NAMCO Studios



※今回のために一部
改修してあります

- AIGameManager
 - ゲーム全体の管理を行っている
- TankManager
 - 戦車1台の各種コンポーネントを管理している
- TankMovement
 - 戦車の移動処理を担っている
- TankShooting
 - 砲弾の処理を担っている
- TankHealth
 - 戦車の耐久力の管理とその表示処理を担っている

◀ TANKS! を改良しよう



BANDAI NAMCO Studios

TANKS!はUnityを学習するための最適なアセットの1つではありますが、残念ながらプログラムコードはお世辞にも良いデキと呼べるものではありません。

- カプセル化がイマイチ
- 責任範囲がおかしい
 - 戦車の出現位置 (TankManager.m_SpawnPoint) は、戦車の外側で管理されるべきモノ
 - そうしないと、ステージを複数用意するなどの拡張に対応できない

逆に、互いのコンポーネントが疎結合なのはとても良い！

良い点はそのまま残して、イマイチなところをリファクタリングしよう。

リファクタリング



BANDAI NAMCO Studios

リファクタリングとは？

- 振る舞いを変えずに、メンテナンスしやすいコードに整理すること
 - 変数名を変えるような些細なことも、構造ごと変えるようなオオゴトも含まれる
 - 『**振る舞いを変えずに**』がとても重要！
 - 振る舞いを変えつつコードを整理すると、整理に失敗した時に大変なことになる！
 - プログラミングに慣れてくるとついやってしまいがちなんだけど、グッと我慢が吉

Visual Studioの代表的なリファクタリング支援機能（C#の場合）

機能	ショートカットキー	メニュー
名前の変更	変えたい名前の上で Ctrl + R, Ctrl + R	【編集】 → 【リファクター】 → 【名前の変更】
メソッドの抽出	コードを選択して Ctrl + R, Ctrl + M	【編集】 → 【リファクター】 → 【メソッドの抽出】
引数の並び替え（署名の変更）	メソッド名の上で Ctrl + R, Ctrl + O	【編集】 → 【リファクター】 → 【パラメータの順序変更】



リファクタリング：実践（1/4）



BANDAI NAMCO Studios

TankManager.cs を ちょっとだけリファクタリングしてみよう！

フィールド（メンバ変数）名に **m_** が付いてるのはダサいので取り除こう！

- m_PlayerColor
 - m_SpawnPoint
 - m_PlayerNumber
- etc...

💡ワンポイント

編集の置換機能を使ってテキストを置き換えると、予想してないものまで置き換わったり、置き換えが不十分になることがあるので、**必ずリファクタリング支援機能を使おう！**

例えば m_SpawnPoint は GameManager でも参照しているので、GameManager.cs も置換しないとビルドエラーになってしまう！

◀ おまけ：m_ はダサい？



BANDAI NAMCO Studios



フィールド名に m_ を付けるのはダサいと言ったな、アレはウソだ。

確かに、Microsoft が定めている規約にも「**プレフィックスは付けるな**」と記されています。

規約は遵守すべきですが、m_ があることでこれはフィールドなんだな・・・と一目でわかることには価値があります。
規約で決まっているから、と頭ごなしに m_ を否定する方がダサい、と私は考えます。



リファクタリング：実践（2 / 4）



BANDAI NAMCO Studios

GameManager と TankManager の役割/責任範囲を考慮したリファクタリングをしてみよう！

- GameManager.SpawnAllTanks メソッド内の for 文のブロックを、別メソッドに分離しよう！
 - 分離したメソッド名は SetupTank にしよう
 - **メソッドの抽出**を使えばメソッド名を入れるだけでうまくいくぞ！

```
private void SpawnAllTanks()
{
    // For all the tanks...
    for (int i = 0; i < m_Tanks.Length; i++)
        SetupTank(i);
}
```




リファクタリング：実践（3 / 4）



BANDAI NAMCO Studios

GameManager.SetupTank メソッドの操作対象のオブジェクトが全て TankManager のインスタンスであることに違和感がある！

```
private void SetupTank(int i)
{
    m_Tanks[i].m_Instance =
        Instantiate(m_TankPrefab, m_Tanks[i].m_SpawnPoint.position,
            m_Tanks[i].m_SpawnPoint.rotation) as GameObject;
    m_Tanks[i].m_PlayerNumber = i + 1;
    m_Tanks[i].Setup();
}
```

本質的に、この処理は戦車のための処理なのだから、SetupTank メソッドの内容は TankManager にあるべき！なので TankManager に移そう！



リファクタリング：実践（3 / 4）



BANDAI NAMCO Studios

GameManager.SetupTank メソッドの内容を TankManager に移そう！

1. TankManager に 以下のようなメソッドを追加して、

```
public void Spawn(GameObject tankPrefab, int playerNumber)
{
    m_Instance = GameObject.Instantiate<GameObject>(tankPrefab, SpawnPoint);
    m_PlayerNumber = playerNumber;
    Setup();
}
```

2. GameManager.SpawnAllTanks を以下のように修正しよう！

```
private void SpawnAllTanks()
{
    for (int i = 0; i < m_Tanks.Length; i++)
        m_Tanks[i].Spawn(m_TankPrefab, i + 1);
}
```



リファクタリング：実践（4 / 4）



BANDAI NAMCO Studios

カプセル化を進めよう！

これまでのリファクタリングの結果、TankManager の

- Setup メソッドは、**外部からアクセスされなくなった！**
⇒ Setup メソッドを public から private に変更しよう！
- m_Instance、m_PlayerNumber フィールドは外部から**変更**されなくなった！
⇒ ゲッターだけのアクセサ（プロパティ）に変更しよう！

```
[HideInInspector] public int m_PlayerNumber;  
↓  
public int PlayerNumber { get; private set; }
```

HideInInspector 属性はUnityエディタ（インスペクター）上で見えなくするだけなので、カプセル化の効果はない。プロパティにすることも、Unityエディタから見えなくすることができるので、今回のケースでは一石二鳥（隠蔽とエディタでの編集不可）。

リファクタリング：名前変更する時のUnity裏技



BANDAI NAMCO Studios

publicなフィールドや、SerializeField 属性が付いたフィールドは Unityエディタで値を設定/変更することができるが、そのようなフィールドの名前を VisualStudioで変更すると、Unityエディタで設定した値が消えてしまう！

しかし、名前を変更する前に **FormerlySerializedAs** 属性を付けておけば、フィールド名を変更してもUnityエディタで設定した値が消えなくなる！

```
public class GameManager : MonoBehaviour
{
    [UnityEngine.Serialization.FormerlySerializedAs("m_Tanks")]
    public TankManager[] m_Tanks;
}
```

こうしておくと、 m_Tanks の名前を変更しても安心なのだ！

LINQ (Language Integrated Query) を使うと、コードを簡潔に書けるようになる！

これが、

```
private void SetCameraTargets()
{
    Transform[] targets = new Transform[m_Tanks.Length];
    for (int i = 0; i < targets.Length; i++)
    {
        targets[i] = m_Tanks[i].m_Instance.transform;
    }
    m_CameraControl.m_Targets = targets;
}
```

こうなる

```
private void SetCameraTargets()
{
    m_CameraControl.m_Targets =
        m_Tanks.Select(tank => tank.m_Instance.transform).ToArray();
}
```


コレクションの拡張メソッドでより便利に



BANDAI NAMCO Studios

```
public static class CollectionExtensions
{
    /// コレクション(シーケンス)の各要素に対してインデックスを付けて指定処理を実行する
    public static void ForEach<T>(this IEnumerable<T> source, Action<T, int> action)
    {
        int index = 0;
        foreach (var item in source) action(item, index++);
    }
}
```

この拡張メソッドを使うと、GameManager.SpawnAllTanks をこう書ける

```
{
    for (int i = 0; i < m_Tanks.Length; i++)
        m_Tanks[i].Spawn(m_TankPrefab, i + 1);
}
```

```
{
    m_Tanks.ForEach((tank, i) => tank.Spawn(m_TankPrefab, i + 1));
}
```

TANKS! AIバトルの準備



BANDAI NAMCO Studios



爆音が鳴るかもしれないのでPCの音量
設定で調整してね

- 作業環境構築
デスクトップにある以下のバッチファイルを実行！
CopyUnityProject.bat
途中で戦車番号を聞かれるので自分の番号を入力！
 - もしデスクトップにバッチファイルがなかったら、以下にあるのでデスクトップにコピーしてください。
¥¥172.28.35.250¥Intern2019¥programmer
- Unity Hub を起動してプロジェクトを登録
 - プロジェクト⇒リストに追加 を選択
 - マイドキュメントの中の Tanks を指定
 - Unityバージョンを選択し、プロジェクトをアップグレードする

TANKS! AIバトルのAI実装



BANDAI NAMCO Studios

マイ戦車のひな形を作成しよう！

- Assets¥Scripts¥TankAI¥TankAI-Sample フォルダの中にある TankAI_00.cs を自分の戦車AIフォルダにコピーして、ファイル名とクラス名を自分の戦車番号に変えよう。

マイ戦車の名前と色をオーバーライドしよう！

- 名前と色は自由に変えられる！
 - 変更の仕方は TankAIBase クラスを見ればわかる！
 - 明るめの色がオススメ！

仮想敵はキーボード操作で移動＆攻撃可能！

- W：前進、A：左旋回、D：右旋回、スペース：発射



よくありそうな質問



BANDAI NAMCO Studios

Q：起動したけど何も出ない！

A：慌てず騒がずシーンをロードしよう！ Assets/_Complete-Game

Q：全員のステータスが見えない！

A：ゲーム画面サイズを 1920 x 1080程度にして！

Q：戦車AIクラス以外を修正していいの？

A：修正しても、ホストPCのコードは修正されないので意味がないです。

Q：仮想敵として他の戦車AIを登場させたい！

A：サンプルAIをコピーして、クラス名の数字を適切（01～22）に変えれば可能です。他人のAIをパクる裏技もあるけど、誰かが気付くまでは公開しませんw