#1
C#と時の部屋
～メソッドとデバッグ編 *int bool if for*～

旅人「C#入門だ！う…、データ型とかクラスとか
いきなりワケワカメや＼(^o^)／！！！」
C#「intが分かればええよ。ifとforで夢が広がるんや。」

ご注意

- Windows環境を基本に説明しています。
- 独断と偏見でピックアップした内容です。
- 初めての人向けに、内容を絞っています。
  堅実な方法を重視しています。
- 初めての人は、慣れてから覚えれば良い
  内容は無視しても良いです。

今回の目的

- 開発環境を準備してデバッグする！
- メソッドを書いて実行する！
- 変数で int(整数型) と bool(論理型) を使う！
- if で条件判定する！
- for で繰り返し(ループ)処理する！

開発環境

- 開発環境が強力であることも、C#の良さです。

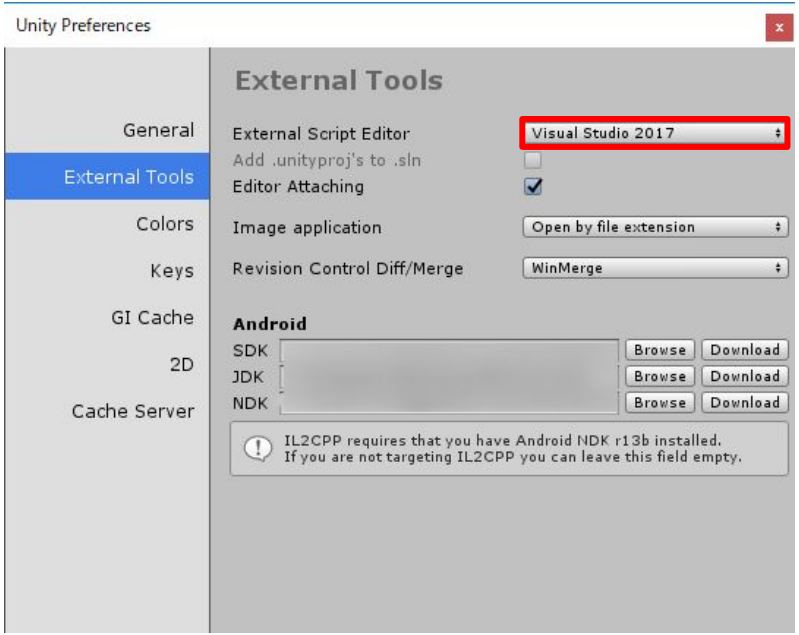
- MonoDevelopやXamarin Studioは捨てましょう。
  https://blogs.unity3d.com/jp/2018/01/05/discontinuing-support-for-monodevelop-unity-starting-in-unity-2018-1/

- Windows
  - Visual Studio Community 2017 (Visual Studio Tools for Unity)
  - http://tsubakit1.hateblo.jp/entry/2016/11/20/233000

- Mac
  - Visual Studio for Mac
  - http://kan-kikuchi.hatenablog.com/entry/VisualStudioforMac

開発環境

1. [Preference…]を開き、Visual Studio に設定する。



開発環境

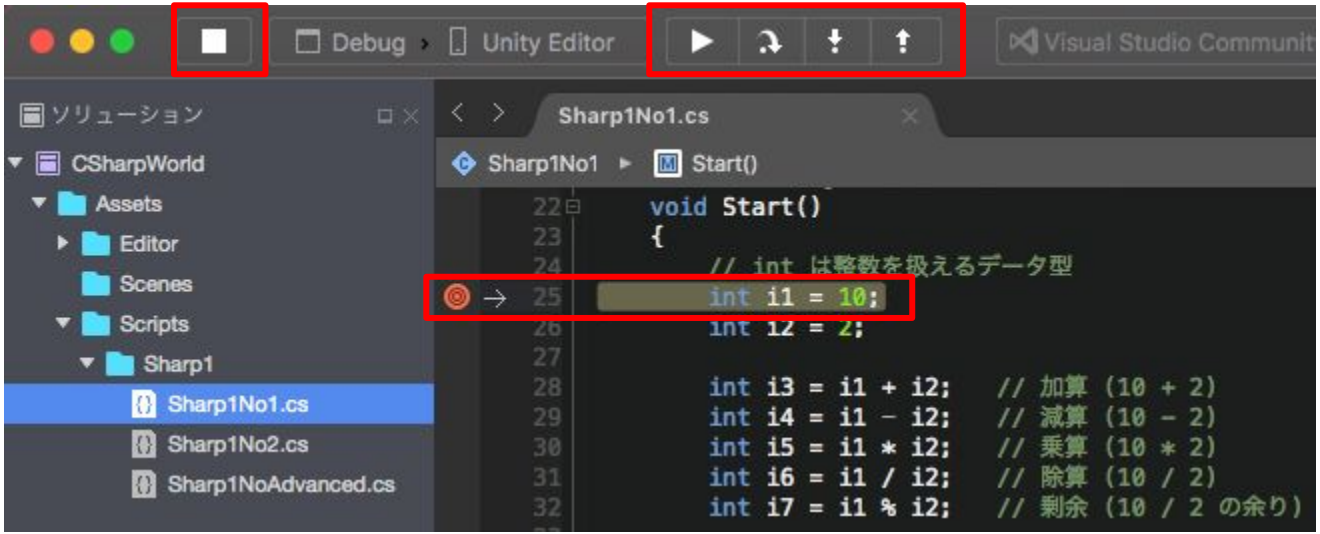
2. [Project Settings] -> [Player] から [Player Settings] を開く。

3. Scripting Runtime Version を .NET 4.x に設定する。

現在は C# 6.0 / .NET 4.7.1 (Unity)
最新は C# 7.3 / .NET 4.7.2
https://blogs.unity3d.com/jp/2018/03/28/updated-scripting-runtime-in-unity-2018-1-what-does-the-future-hold/

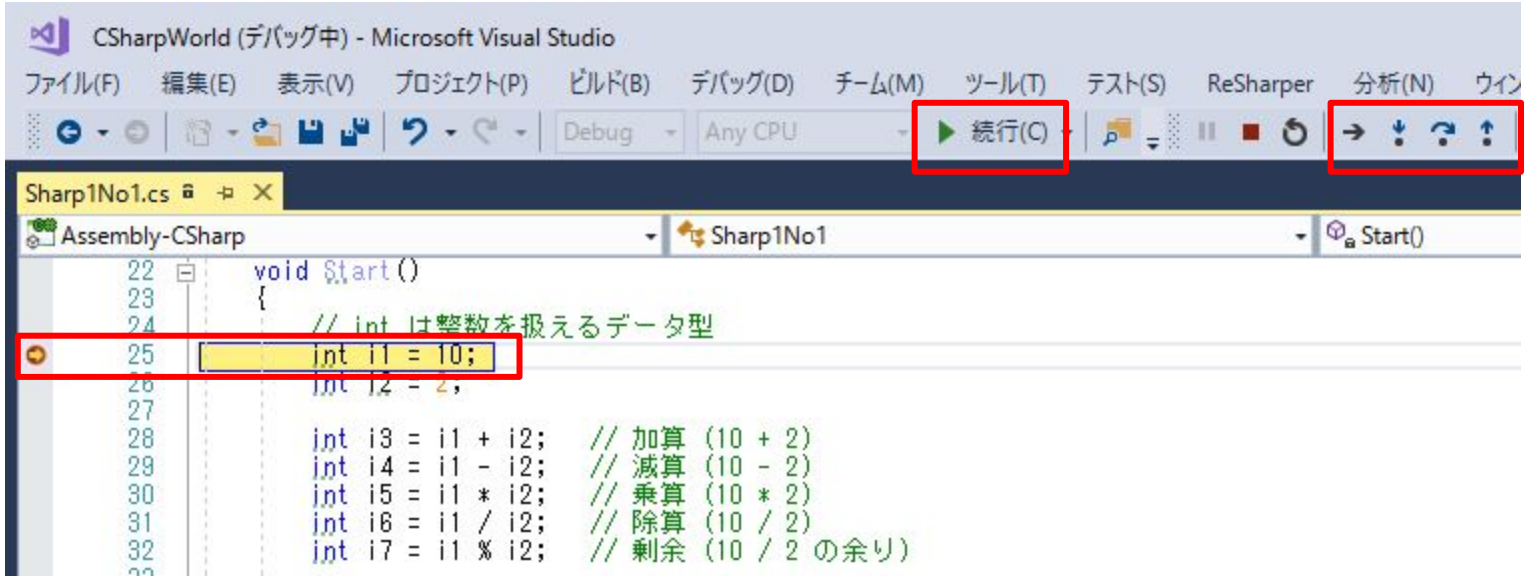
デバッグ

1. ブレイクポイント(一時停止できる)を設定する。
2. Visual Studio(以下VS)で、Unityにアタッチする。
3. Unity を Play してデバッグできるか確認する。



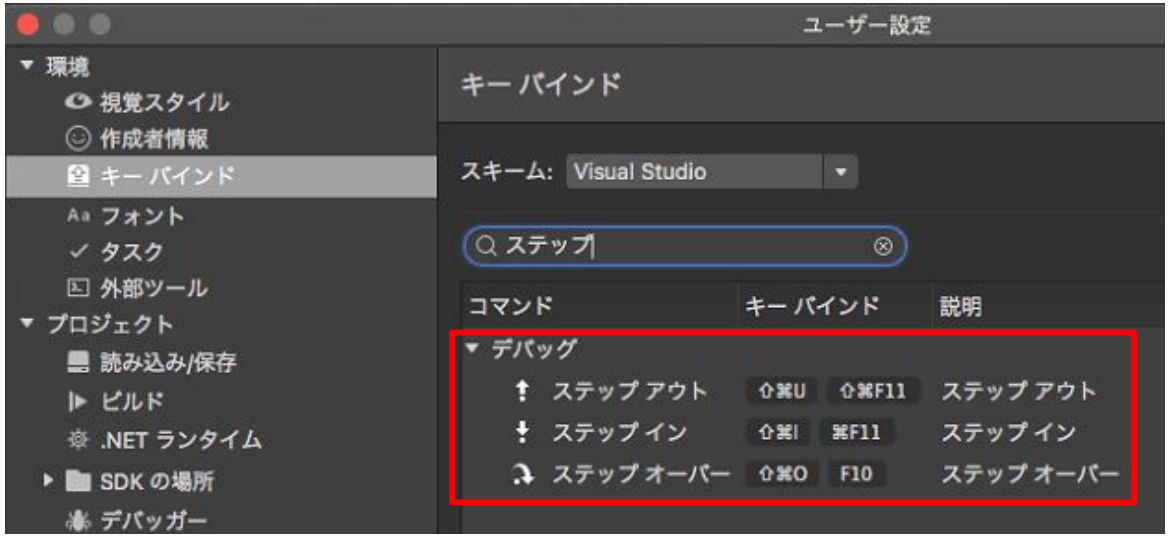
デバッグ

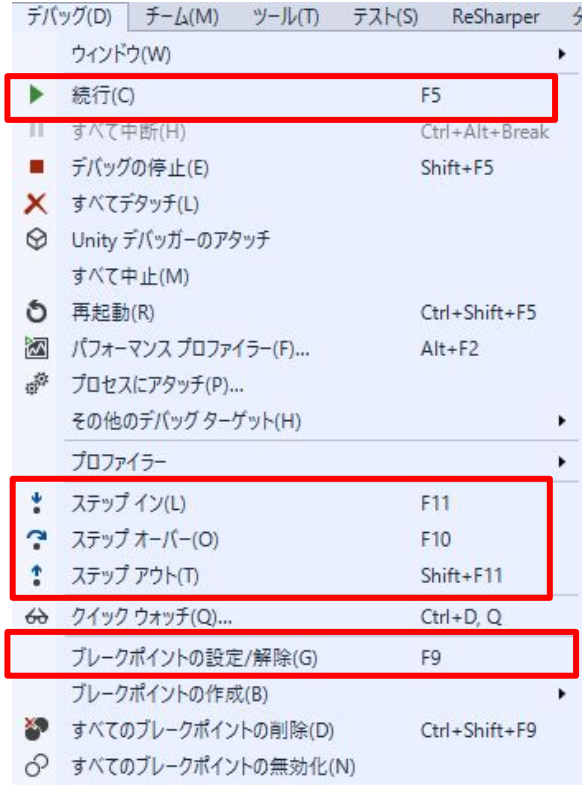
4. Windowsの場合。



デバッグ

5. ショートカットキーを確認する。





ソースコード1

Sharp1シーンを開き、No1をアクティブにして確認しましょう。

1. **コメント** … 実行されないコード。
2. **int** … 整数を扱うという宣言。
3. **変数** … 名前を付けて整数を入れておく。
4. **演算** … 四則演算やインクリメントなど。
5. **見やすく** … 途中計算を変数に入れたり、改行する。
6. **メソッド** … 処理のひとかたまり。
7. **戻り値** … メソッドが返す値。
8. **引数** … メソッドに渡す値。
9. **return** … メソッドを抜ける。戻り値を返す。
10. **警告** … 未使用の変数などは解消する。

ソースコード2

No1を非アクティブにして、No2をアクティブにして確認しましょう。

1. **bool** … 論理値 true か false の2択。
2. **if** … 論理値による条件判定。
3. **&&** … 両方を満たすAND条件。
4. **||** … 片方を満たすOR条件。
5. **演算順序** … 左から右へ、&& が先、|| が後。
6. **? :** … ifを使わない三項演算子。
7. **for** … ループ処理。
8. **contirue** … 次のループへ。
9. **break** … ループを抜ける。
10. **多重ループ** … メソッドを分割して対応する。

ソースコード3

No1を非アクティブにして、No2をアクティブにして確認しましょう。

1. **System.Int32** … intのエイリアス。
2. **var** … 右辺から型推論。
3. **最適化** … リリースビルドは最適化される。
4. **インクリメント** … 前置と後置。
5. **if else** … { } の省略は注意して。
6. **短絡評価** … & や | は通常使わない。
7. **ifの判定** … boolを渡せば、== trueは不要。
8. **if内の代入** … 条件式内で変数に代入しない。
9. **unchecked** … intは最大値を超えると最小値になる。
10. **checked** … オーバーフロー時に例外になる。

余談

初心者プログラマが犯しがちな過ち25選
(初心者が読むには難しいけど、面白い内容を紹介)

https://qiita.com/rana_kualu/items/379eefb3a40c6b44cb92#24-%E3%82%A8%E3%83%A9%E3%83%BC%E3%82%92%E5%AB%8C%E3%81%86

24) エラーを嫌う … エラーは幸せラッキーヒント。
“プロの開発者はエラーを愛しますが、初心者は嫌います。”
ビルドエラーに苦しむ人へ(過去の自分)。エラーのないバグが辛い。

22) バージョン管理しない … バージョン管理はあなたの力になる。
“初心者はバージョン管理システムの力を知りません。”

#1完了
C#と時の部屋
～メソッドとデバッグ編 *int bool if for*～

int, bool, if, for は今後もずっとお友達です。
シンプルなコードを心がけて、
怪我(バグ涙)を回避して仲良く遊びましょう！