

clusterでワールド作りたい人集合!

LOGIC

そるそる

21:00開始予定

やるうか!

じゃあ



[Twitter @vins_cluster](https://twitter.com/vins_cluster)



vinsです。Logicとか多用したワールドが好きです
GameJAM 2020冬では「カンヅメRPG」で
大賞もらいました

Logic勉強会

- 初心者～中級者ターゲット
- 過去のvinsの記事や動画の紹介も多くなると思います

**で、早速過去の勉強会の
繰り返しになりますか**

そもそもCCKの場合、Logicというのは
データ保存機能でもある
HPやMP、鍵とったフラグ、銃の残弾、
ゲーム残り時間、チームの得点、
倒さなきゃいけない残り敵数……

ItemLogic

アイテムにデータを保存する。必要なら
計算もする。必要ならメッセージも送信する。

(例: 銃の残弾管理、敵のHPや行動管理)

PlayerLogic

プレイヤーにデータを保存する。必要なら (ry
(例: HPやMP、EXPやLV、攻撃力や防御力、
現在の速度やジャンプ力、カギとったフラグ)

前に
やった

ItemLogic PlayerLogic GlobalLogicの使い分け

ここの掘り下げは**第1回ゆるゆる勉強会**の

PDFと動画で

(イベントページにリンクを張っています)



とはいえ、もちろん保存だけが
Logicの機能ではない

Logicの
機能は



- 数字を保存する
- 計算する
- 条件チェック

(※「保存」といっても特殊な処理をしない限り、
ワールドを出たら基本的にデータは消える)

では

具体的に

1秒ごとに

違う音が順番に鳴る

Itemをつくります

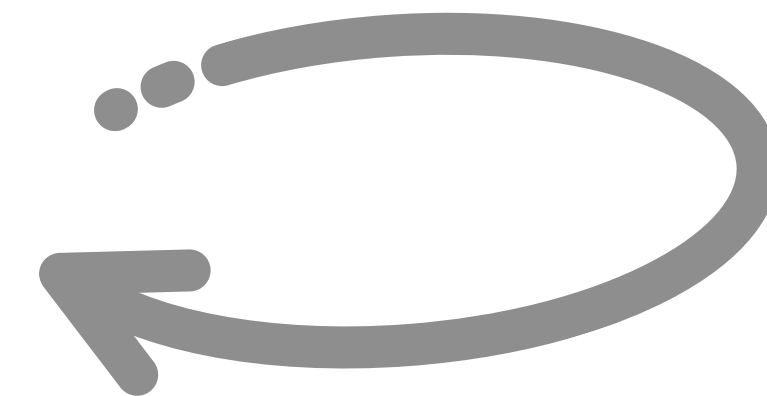
多分初心者向け



OnCreateItemTrigger



↓
ItemTimer



タイマーは
永久ループ

↓
ItemLogic

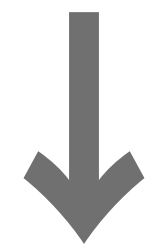
↙
PlayAudio 

SourceGimmick

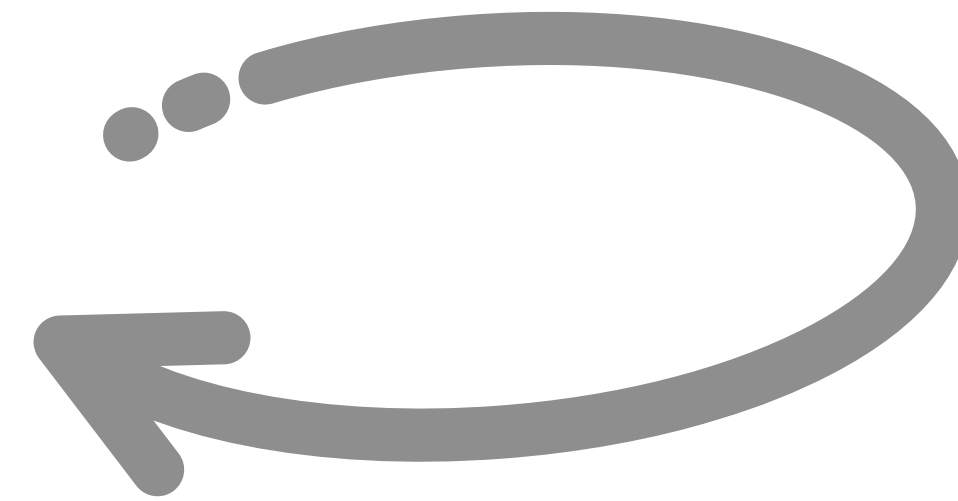
↘
PlayAudio 

SourceGimmick

OnCreateItemTrigger



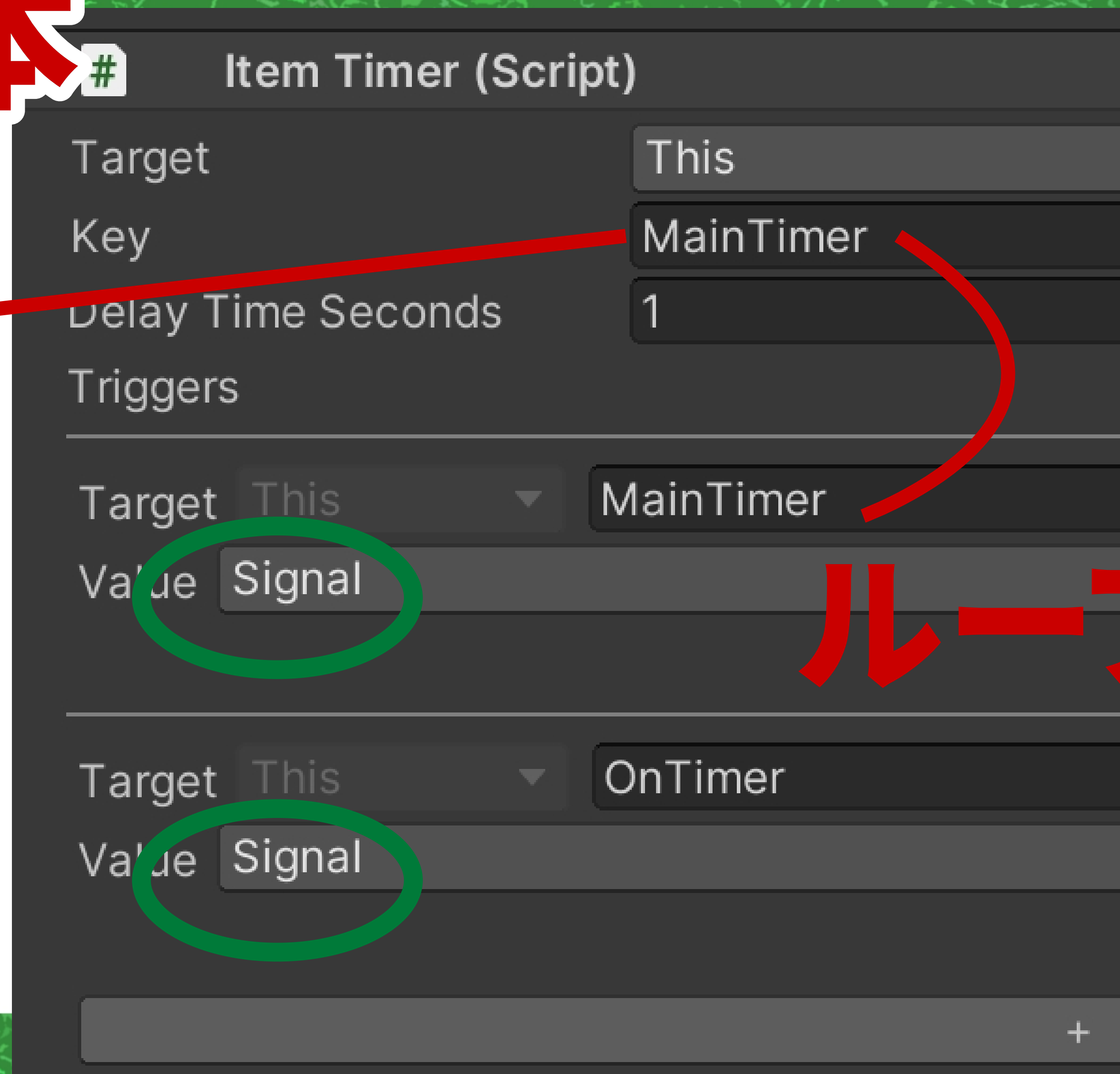
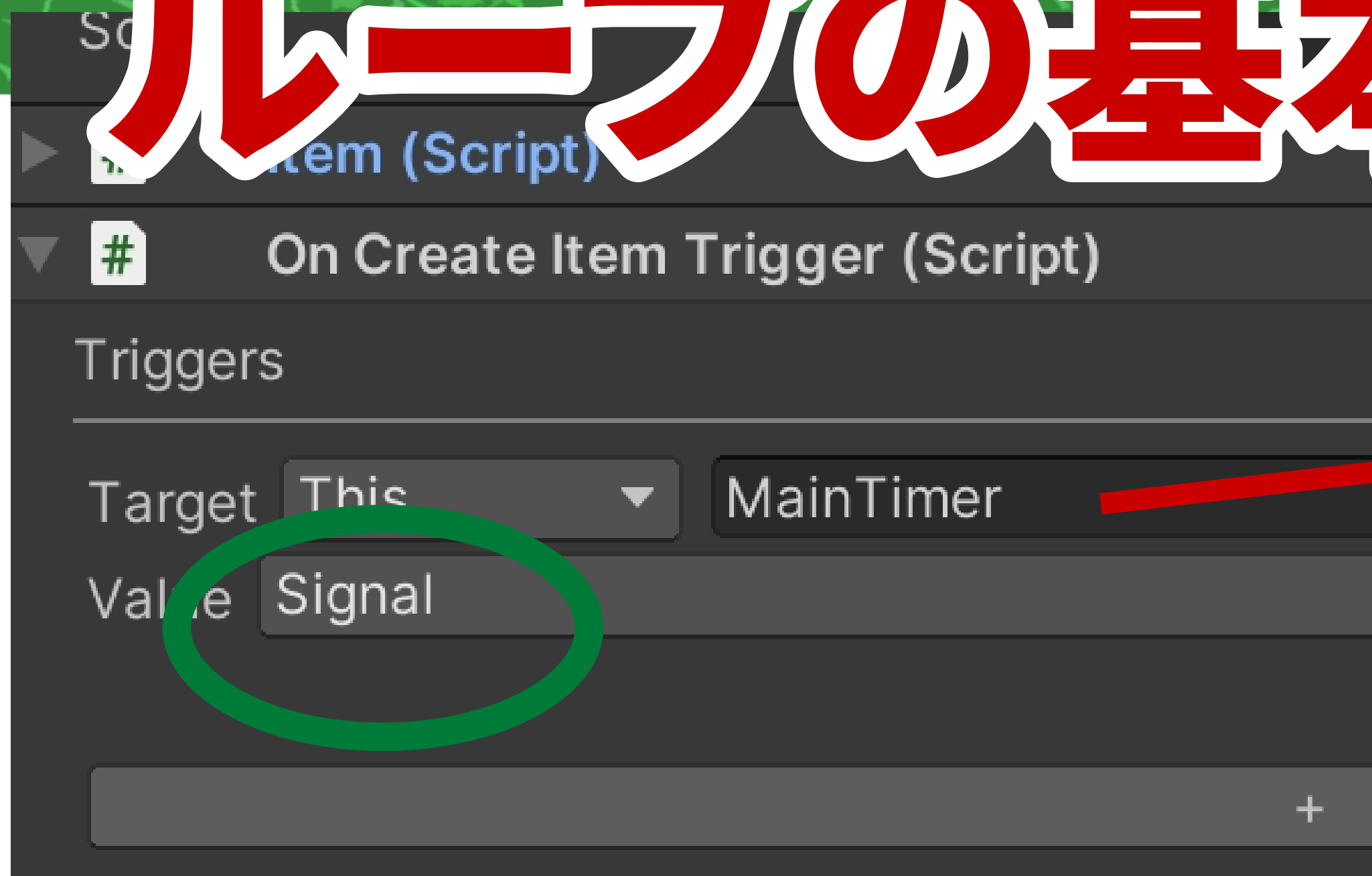
ItemTimer



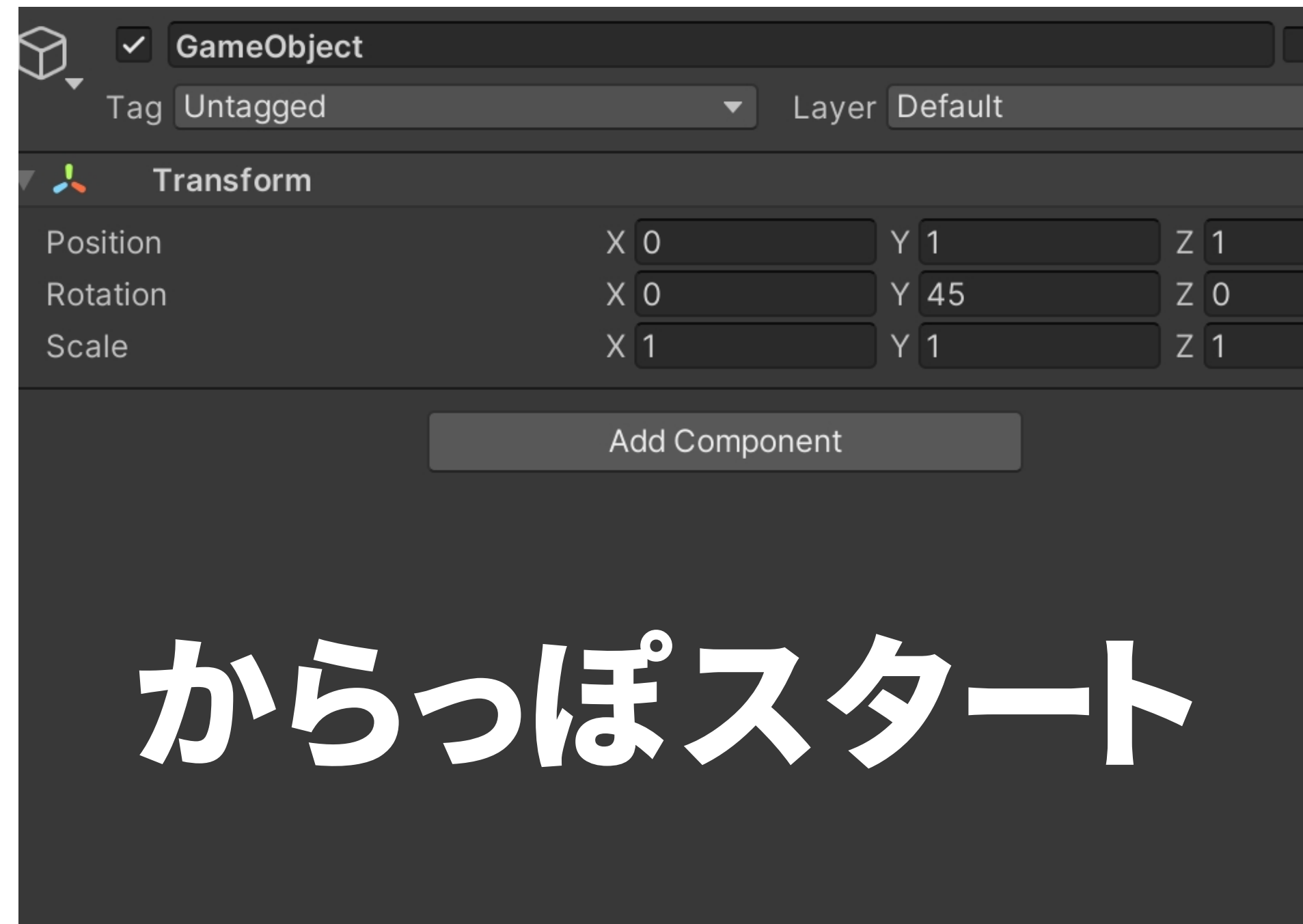
タイマーは
永久ループ

Logicナシでもこれは
使いまくるパターン!

ループの基本



Signalは
「呼び出し」「起動」
と考えよう



からっぽスタート

CCK関係を付ける、からっぽのモノが「親」「見た目」のCapsuleとかは「子」にする これも基本

で、やっとLogicが
出てきます



Item Logic (Script)

Target This

Key OnTimer

This ▼ val Integer

= Add

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 1

This ▼ val Integer

= Modulo

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 3

This ▼ sound0 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 0

This ▼ sound1 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 1

This ▼ sound2 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 2

+



Item Logic (Script)

Target

This

Key

OnTimer

1を足す

This

val

Integer

= Add

RoomState

Integer

This

val

Constant

Integer

1

3で割った余り

This

val

Integer

= Modulo

RoomState

Integer

This

val

Constant

Integer

3

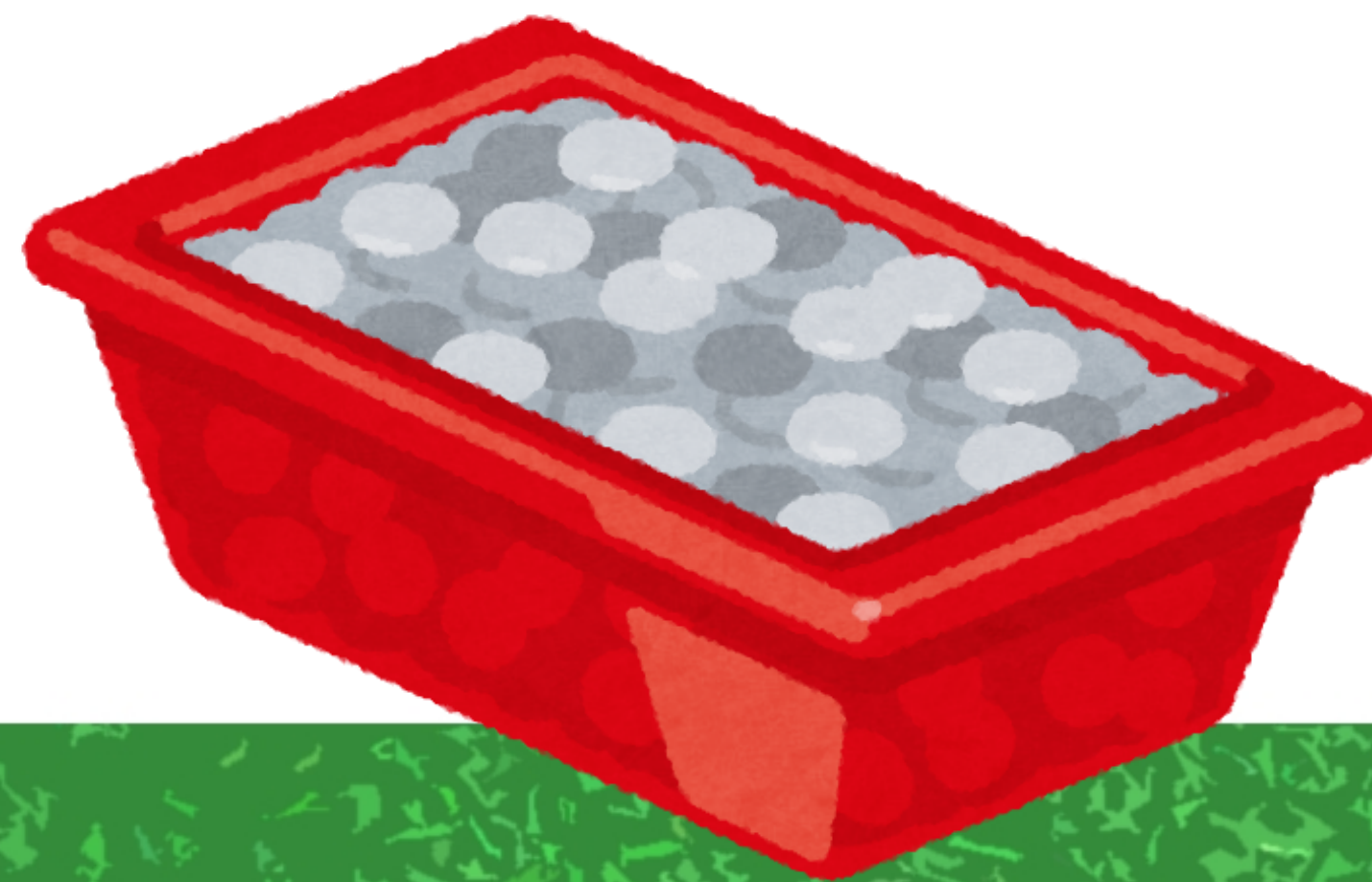
valが

1,2,0,1,2,0,1,2,0...

と変化していく計算!

※初期値は何もしなくても0になります

ちなみに数字を中に入れて
保存するものを
「**変数**」といいます (CCK公式用語ではない)



CCK公式では「メッセージ」と表現されているが……

●Signalは「メッセージ」

●Signal以外は「変数」

……と考えるほうがいいのかも？

(初心者のうちは)

Item Logic (Script)

とりあえず最初は
ぜんぶThisに!

Target This

Key OnTimer

This ▼ val **計算結果を入れる変数は?** Integer

= Add

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 1

This ▼ val **計算結果を入れる変数は?** Integer

= Modulo

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 3

変数

ただの数字

変数

ただの数字

▼ # Item Logic (Script)

Target

This

Key

OnTimer

This ▼

val

Integer

= Add

RoomState ▼

Integer ▼

This ▼

val

Constant ▼

Integer ▼

1

小数じゃない

This ▼

val

Integer

= Modulo

RoomState ▼

Integer ▼

This ▼

val

Constant ▼

Integer ▼

3

小数じゃない

Item Logic (Script)

Target This

Key OnTimer

This ▼ val Integer

= Add

RoomState ▼ Integer

Constant ▼ Integer

This ▼ val Integer

= Modulo

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 3

**結果は
小数点ナシで
変数に保存!**

Item Logic (Script)

Target This
Key OnTimer

This val Integer
= Add
RoomState Integer This val
Constant Integer 1

This val Integer
= Modulo
RoomState Integer This val
Constant Integer 3

This sound0 Signal
= Equals
RoomState Integer This val
Constant Integer 0

This sound1 Signal
= Equals
RoomState Integer This val
Constant Integer 1

This sound2 Signal
= Equals
RoomState Integer This val
Constant Integer 2

+

すいませんまだ
上の2つだけ
なんですよ



This ▼ sound0 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 0

This ▼ sound1 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 1

This ▼ sound2 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 2

でもこれは
ほとんど
3つとも同じ

This ▼ sound0 Signal

= Equals **イコールですか？**

RoomState ▼ Integer ▼ This ▼ **val** **val**は

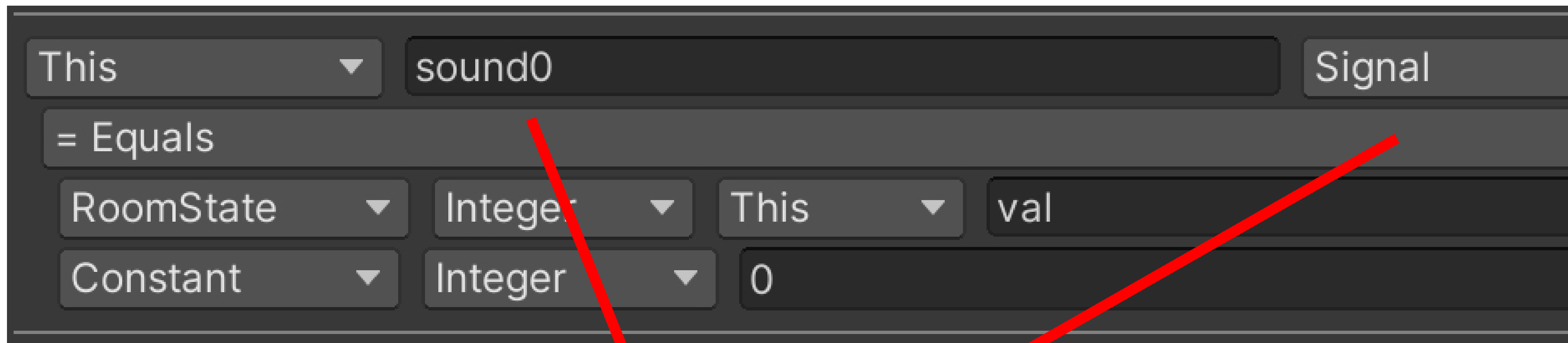
Constant ▼ Integer ▼ 0 **0と！**

変数

ただの数字

さっきから計算してる

valって変数は、いま0ですか？



0のときは、「**sound0**」という
メッセージ (**Signal**) を送りなさい!

This ▼ sound0 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 0

This ▼ sound1 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 1

This ▼ sound2 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ val

Constant ▼ Integer ▼ 2

0のときは～

1のときは～

2のときは～

……と3つ

並んだだけ

The image shows a screenshot of a programming IDE with a dark theme. It displays two sections of code. The top section shows a variable declaration: `val Integer` (where 'Integer' is circled in red), followed by an assignment: `= Modulo`. Below this, there are two lines of code: `RoomState Integer This val` and `Constant Integer 3`. The bottom section shows a variable declaration: `val sound0` (where 'Signal' is circled in red), followed by an assignment: `= Equals`. Below this, there are two lines of code: `RoomState Integer This val` and `Constant Integer 0`.

変数の保存

**メッセージ
の送信**

これでやっと.....



1秒ごとに

「**sound1**」「**sound2**」「**sound0**」の

メッセージを呼び出すLogicができた



音を鳴らすところは？

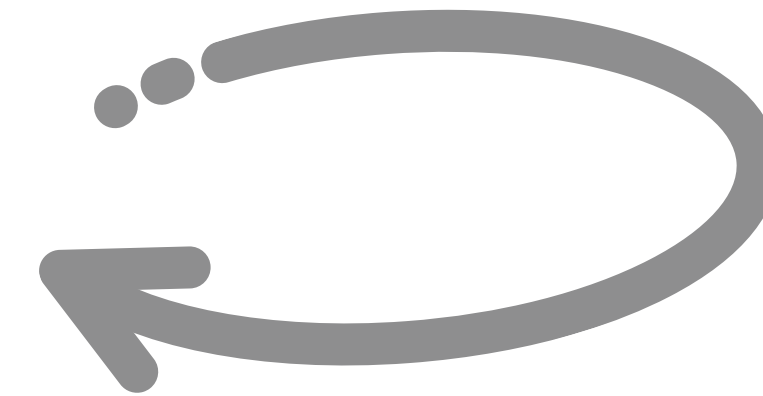
今作るところです



OnCreateItemTrigger



ItemTimer



タイマーは
永久ループ



ItemLogic



PlayAudio



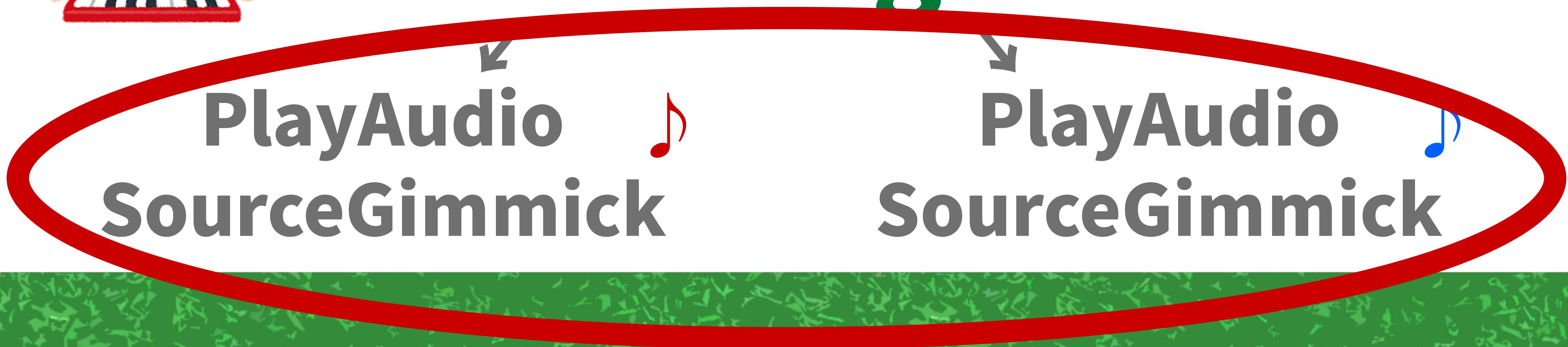
SourceGimmick



PlayAudio



SourceGimmick



(なんとなく名前をOtoObjに変更してみました)



今こんな感じ
PlayAudioSourceGimmickを
直接つけてもいいが.....

音データ

Audio Source **どっち?** OtoObj (Audio Source)

Target Item

Key sound0

Item # OtoObj (Item)

Parameter Type Signal

Audio Source **どっち?** OtoObj (Audio Source)

Target Item

Key sound1

Item # OtoObj (Item)

Parameter Type Signal

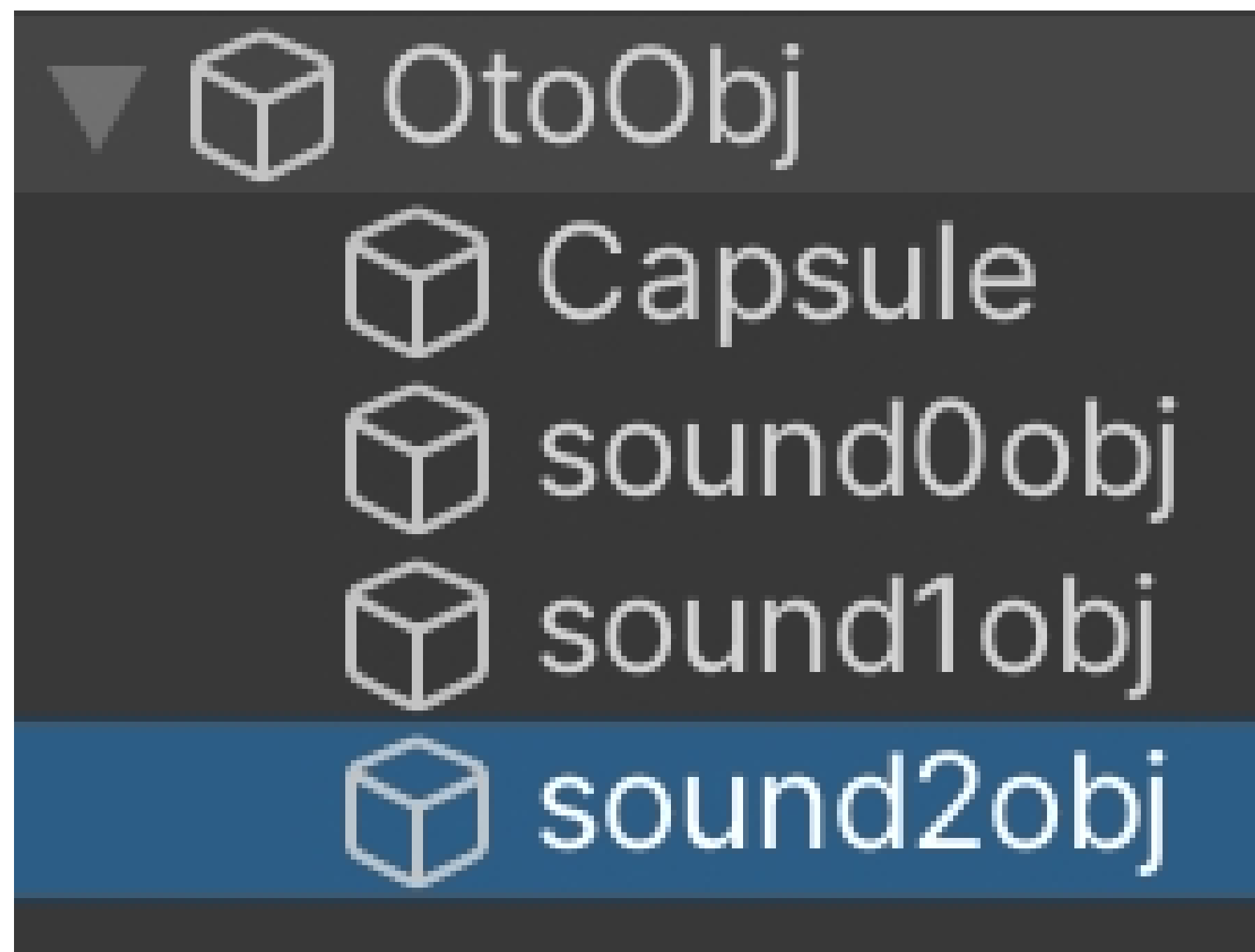
2つ以上
AudioSourceを
同じオブジェクトに
つけると
指定しづらい!

音データ

中級者用? テク



PlayAudioSourceGimmickは
別のオブジェクトに付けてもOK



子にEmptyObjectを
3つ作り適当な名前に

sound0obj
Tag Untagged Layer Defau

Transform
Position
Rotation
Scale

Audio Source

Play Audio Source Gimmick (Script)

Audio Source sound0obj (Audio Source)

Target **Item**

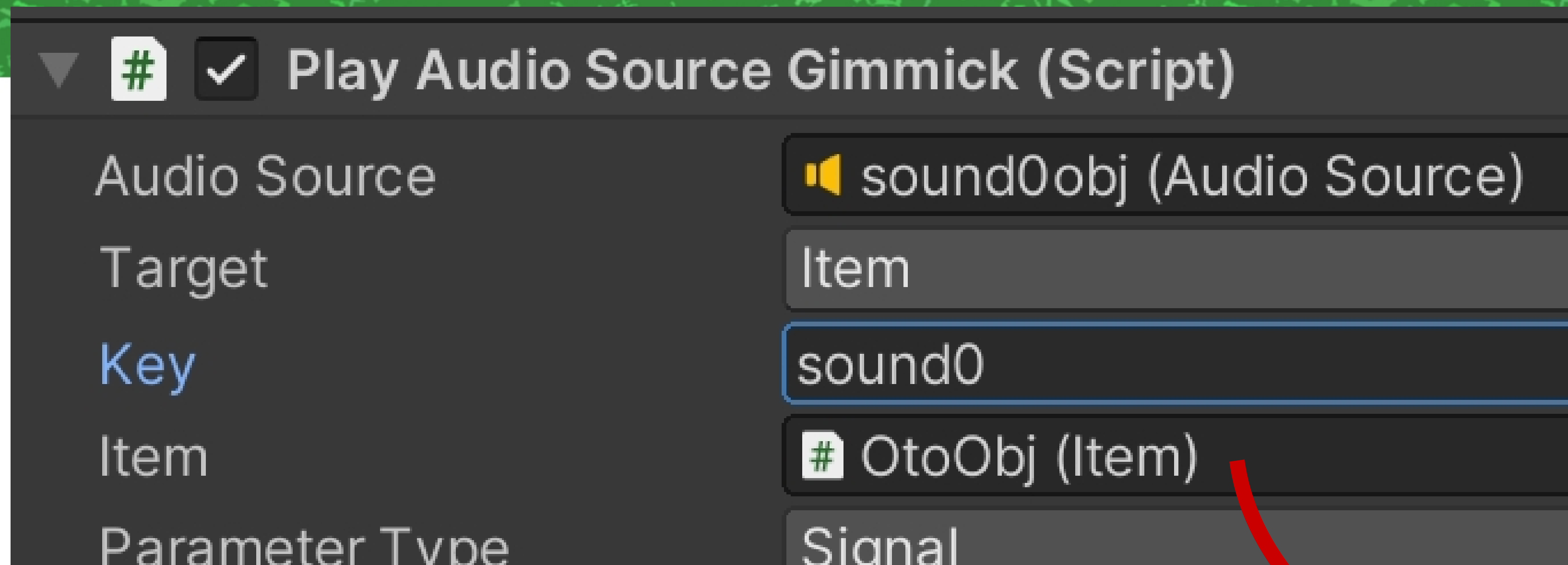
Key

 GimmickTarget を Item にするには Item を指定する必要があります。

Item None (Item)

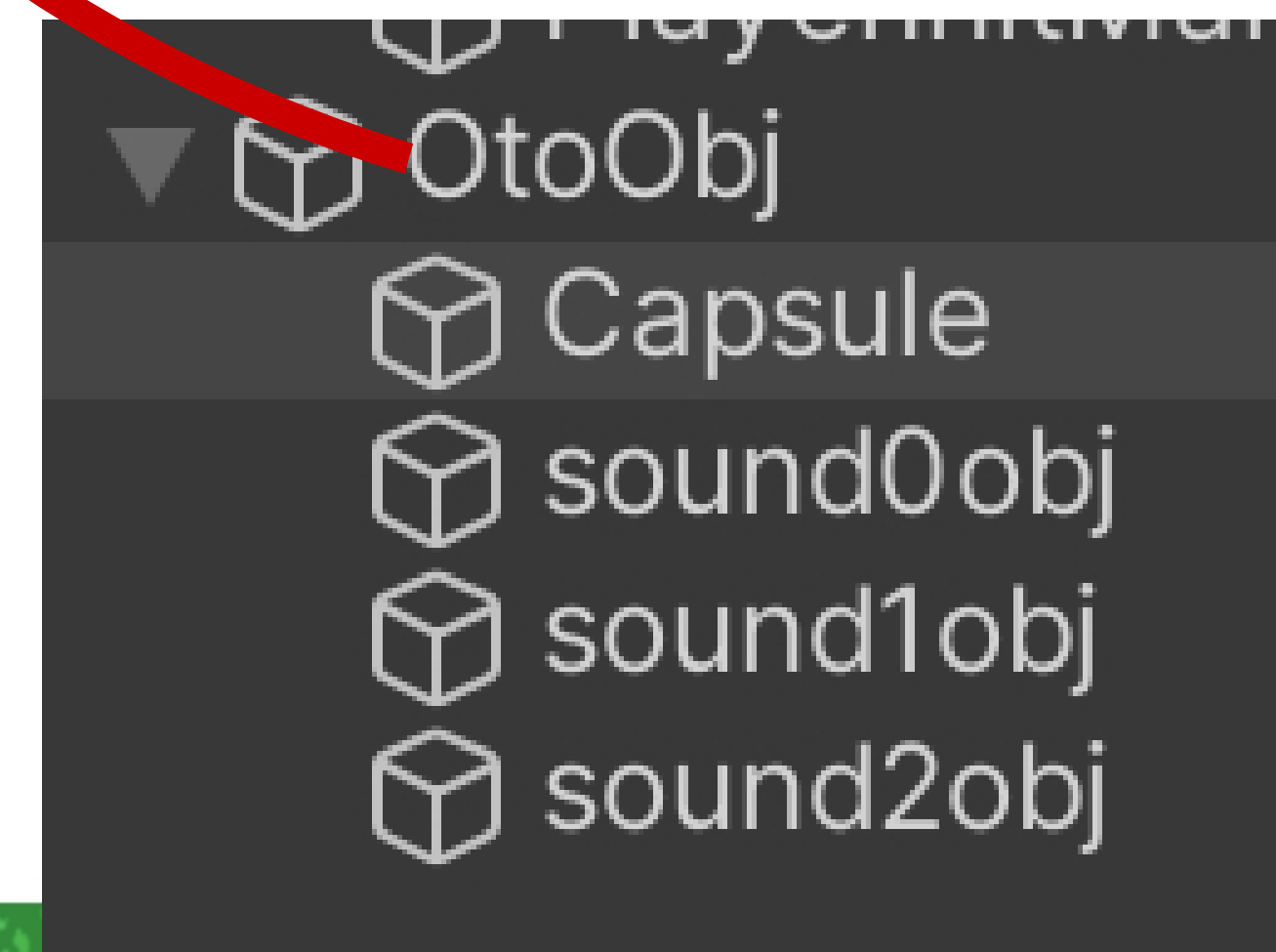
Parameter Type Signal

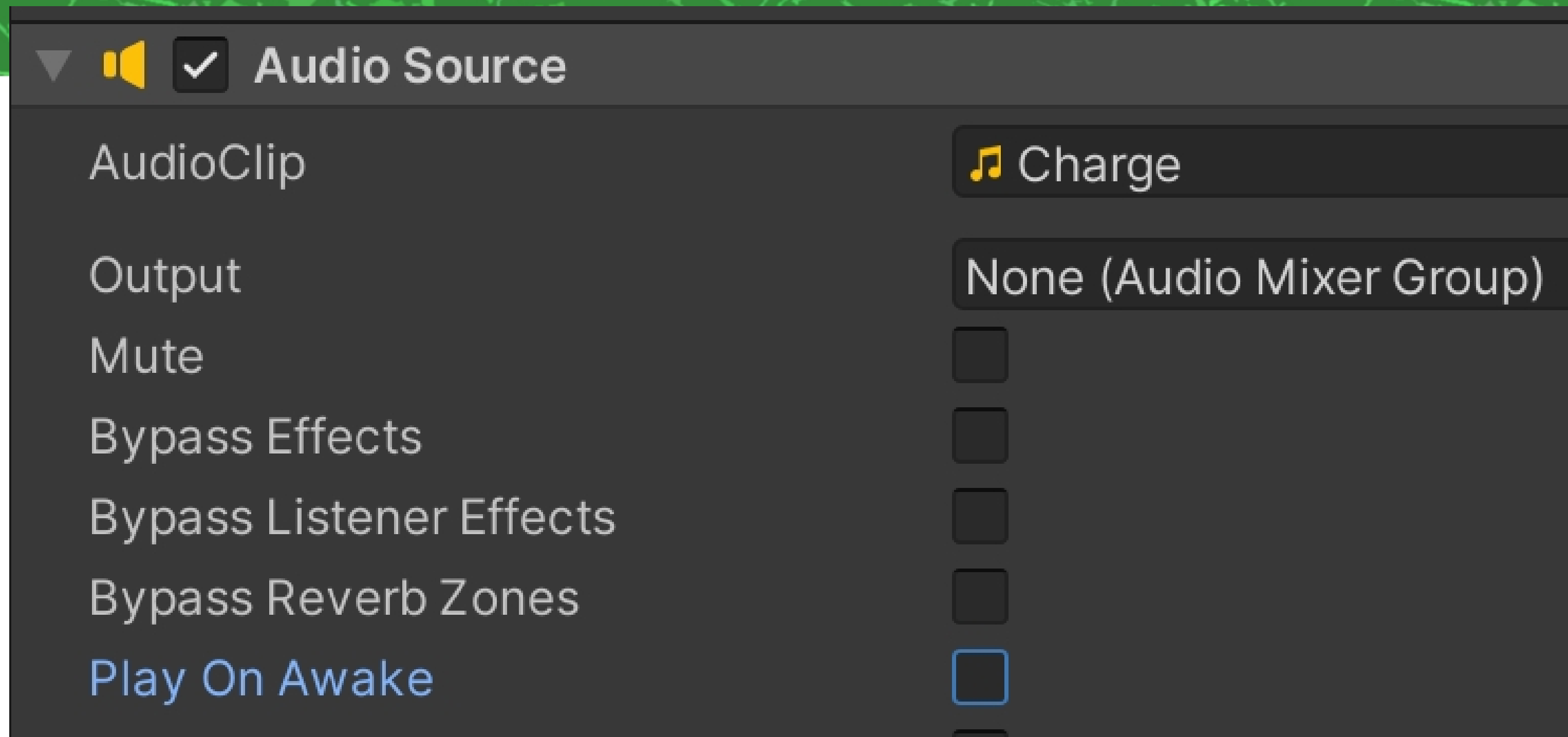
PlayAudioSourceGimmickを
つけ、Targetは「Item」に
するとなんかエラーが出るが



Keyを
sound0に

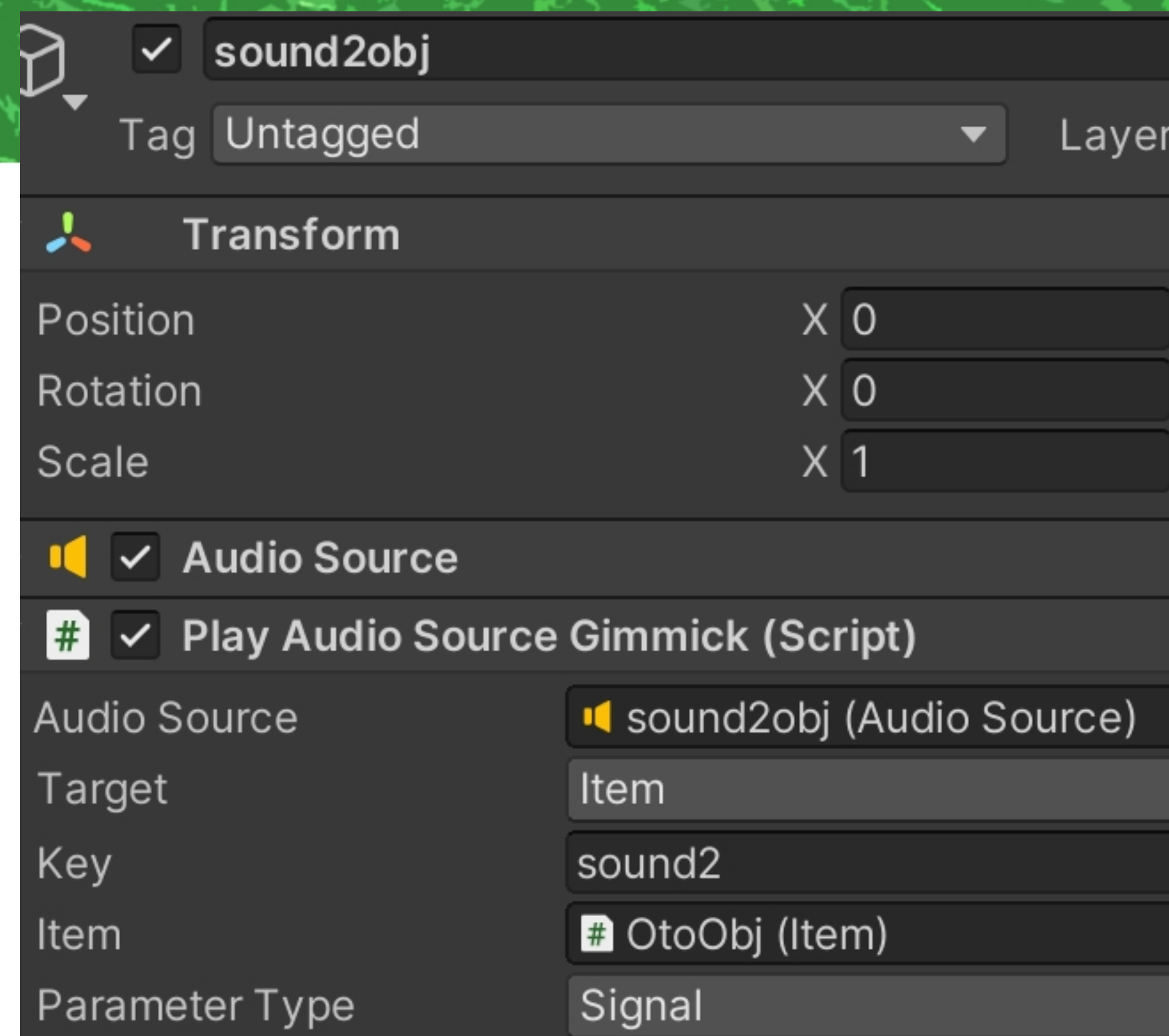
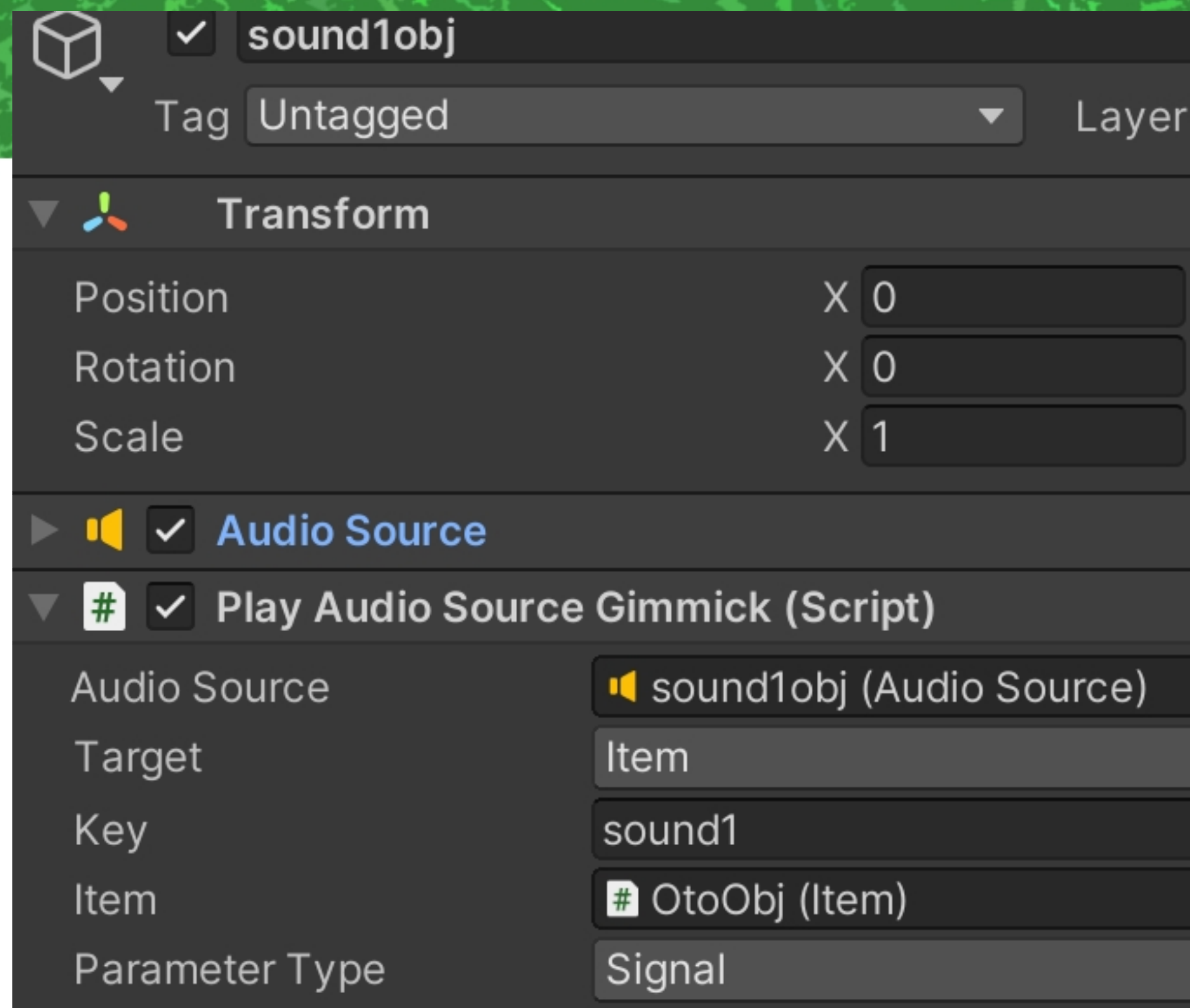
Itemに**OtoObj**を
ドラッグ・アンド・ドロップで
指定すればエラーは消える





今回は
テンプレートワールド
「シューター」から
音を持ってきました

AudioSourceに適切な音を設定、
PlayOnAwakeのチェックは外す



同じことをsound1objとsound2objでもやる
ただKeyは**sound1**、**sound2**にするのを忘れない
あとAudioSourceで別の音を指定



今度こそできた



できるのこれだけ？

坂道を
のぼり始めた
第一歩



前に
やった

Logicでの「計算」一覧

ここの掘り下げはvinsのZennの記事で
(イベントページにリンクを張っています)



前に
やった



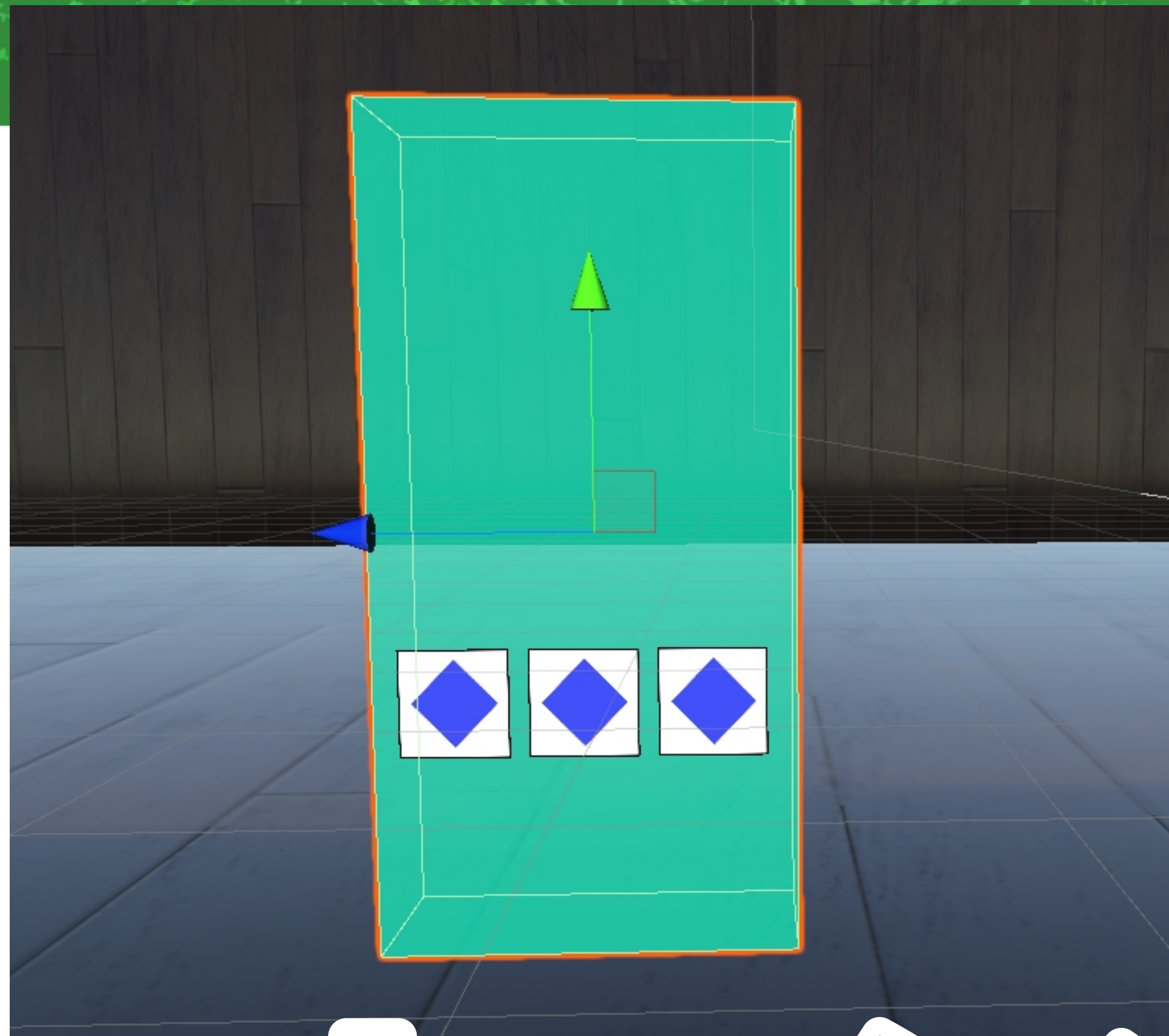
銃の連射速度を制限する例

ここの掘り下げはvinsのZennの記事で
(イベントページにリンクを張っています)





次はやや中級者向けかも
しかも時間ないので
割と駆け足でいきます



スロットマシン風

InteractItemTriggerが起点

Playerのお金減らす

ロール開始・**ItemTimer**設定

SetAnimatorValueGimmickで絵柄変更

InteractItemTriggerされるたび1つ停止

全部停止したら役をチェック

▼ PlayerManagers
PlayerInitManager

EmptyObjectに
PlayerLogicとかを付ける
移動速度やジャンプ力も
ここでやっておく

▼ # Set Move Speed Rate Player Gimmick (Script) ⓘ ⚙ ⋮
Target Player
Key moveSpeed

▼ # Set Jump Height Rate Player Gimmick (Script) ⓘ ⚙ ⋮
Target Player
Key jumpHeight

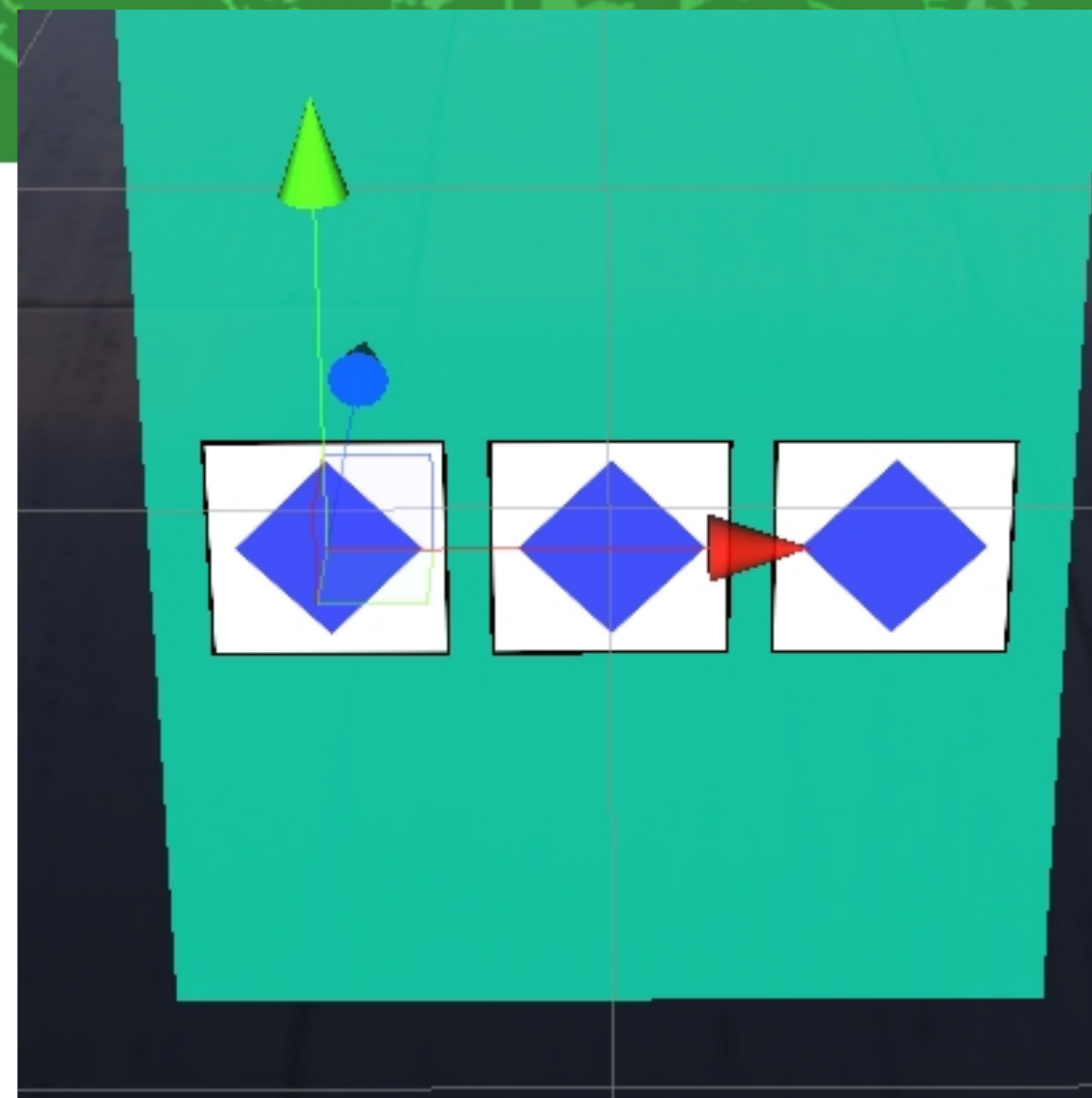
▼ # On Join Player Trigger (Script) ⓘ ⚙ ⋮
Triggers
Target Player OnInit
Value Signal

▼ # Player Logic (Script) ⓘ ⚙ ⋮
Target Player
Key OnInit

Player jumpHeight Integer
=
Constant Integer 3

Player moveSpeed Integer
=
Constant Integer 3

Player Money Integer
=
Constant Integer 30



- Machine
- Cube
- spriteObj00
- spriteObj01
- spriteObj02

spriteObj00 Static

Tag Untagged Layer

Transform

Position X -0.31 Y 0.122 Z 0.3

Rotation X 0 Y 90 Z 0

Scale X 0.1 Y 0.1 Z 1

Sprite Renderer

Sprite sprite00

Color

Flip X Y

Draw Mode Simple

Mask Interaction None

Sprite Sort Point Center

Material Sprites-Default

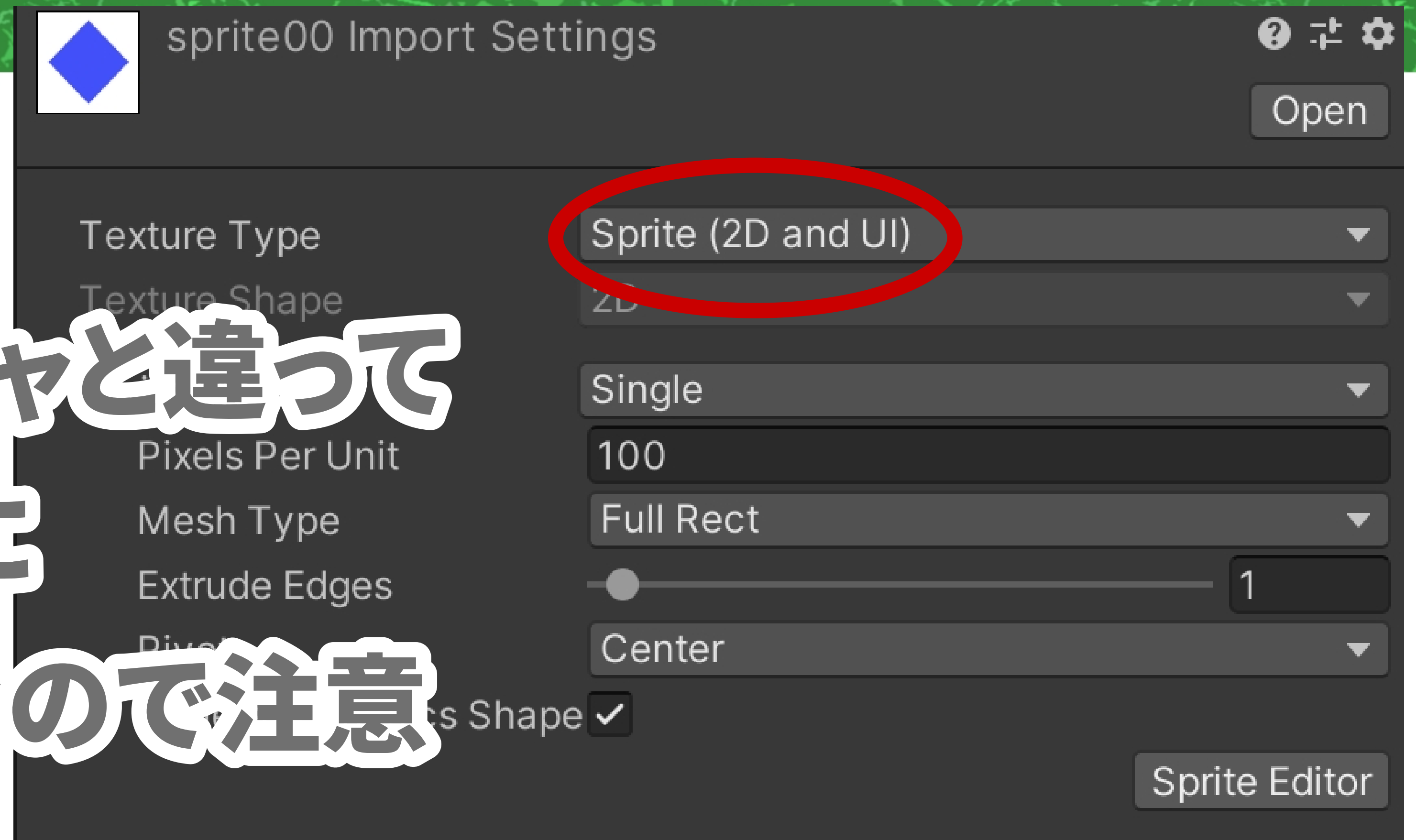
Additional Settings

Sorting Layer Default

Order in Layer 0

ただの箱の前に
3つSpriteを付けた

**Spriteはテクスチャと違って
インポートした後に
設定変更が必要なので注意**

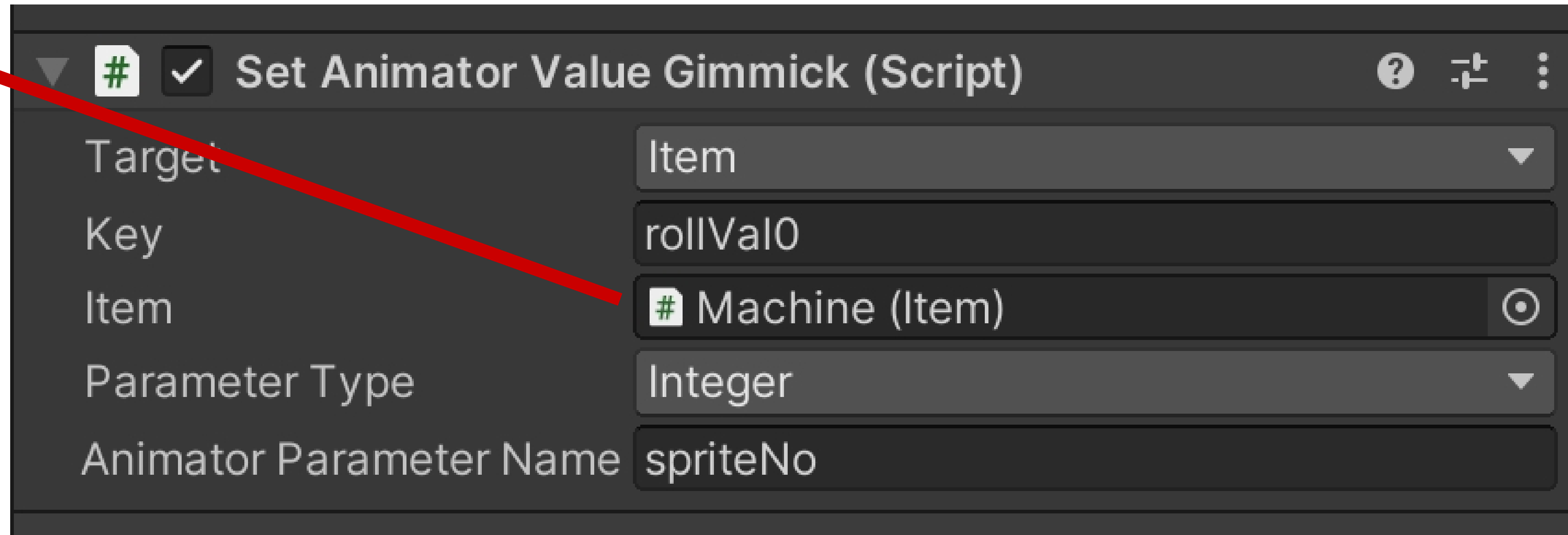
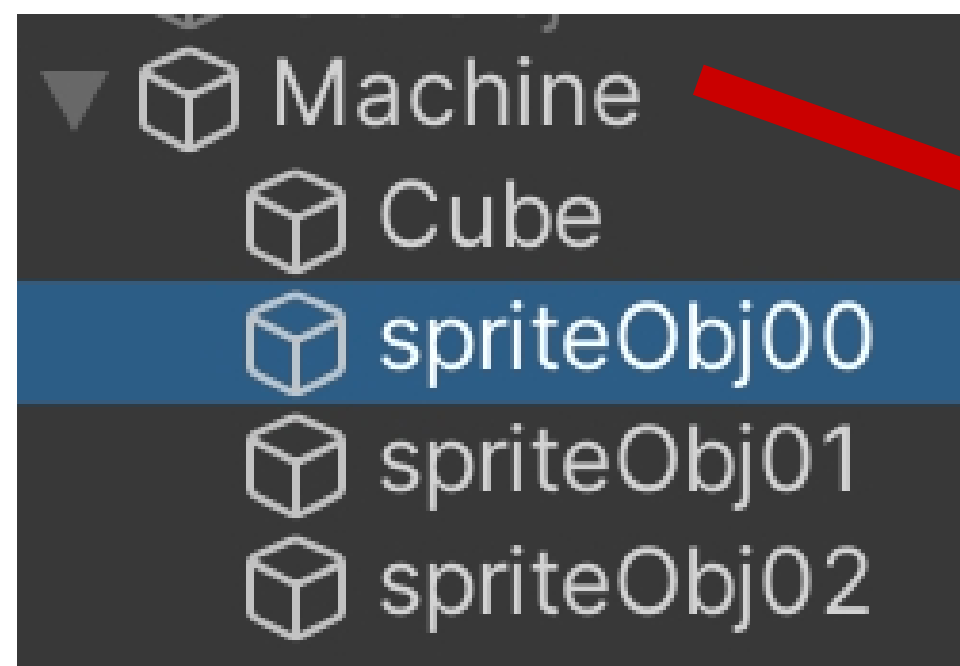


**Sceneにドラッグ・アンド・ドロップすれば
勝手にSpriteRenderer付きます**

Logic関係の先に
Sprite関係の
Animator作ります

長くて細かいので動画。

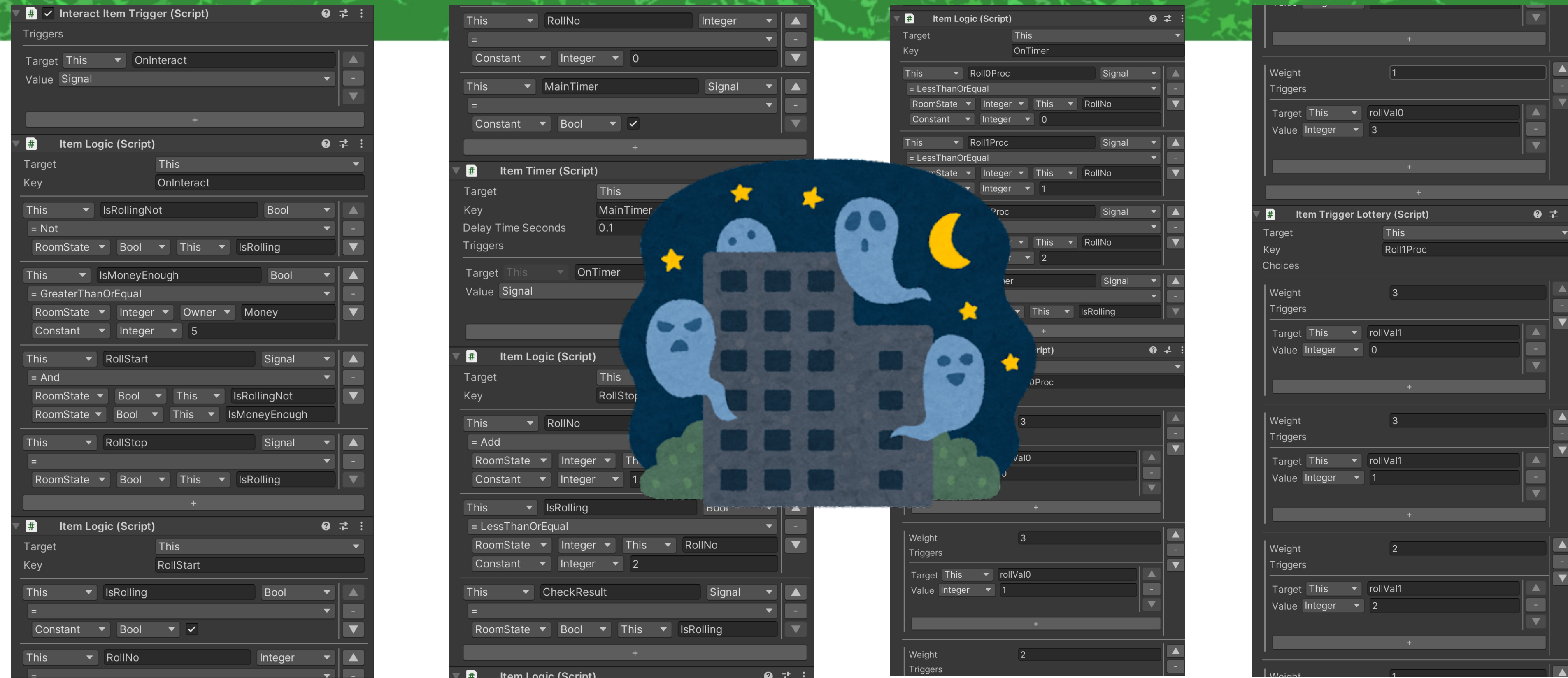




各SpriteにSetAnimatorValueGimmick

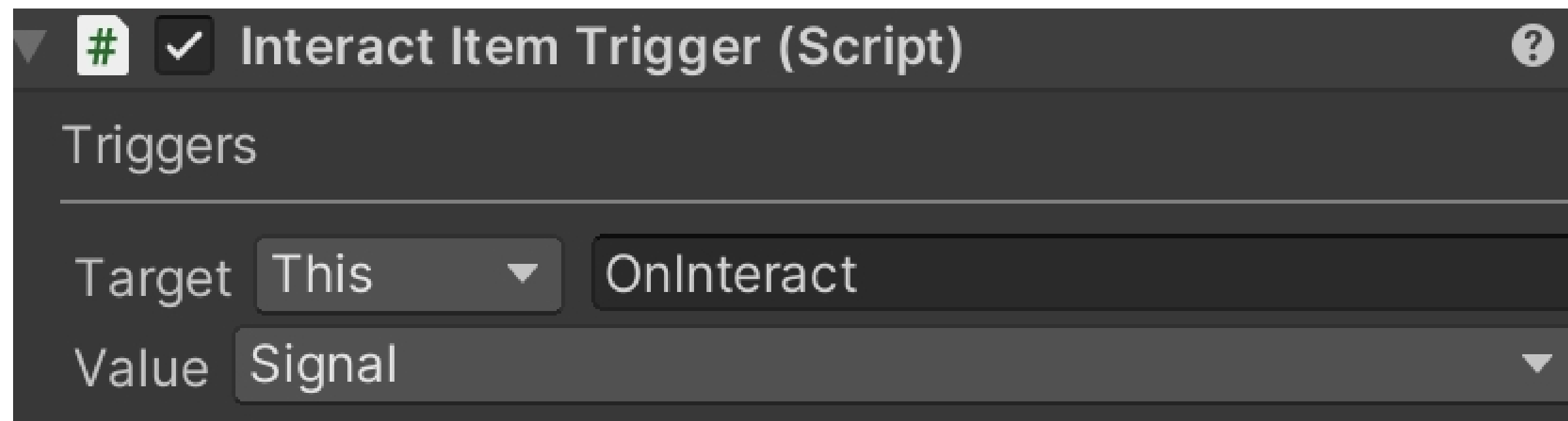
音関係と同じで子にGimmickを付ける。

Machineに
直接つけない

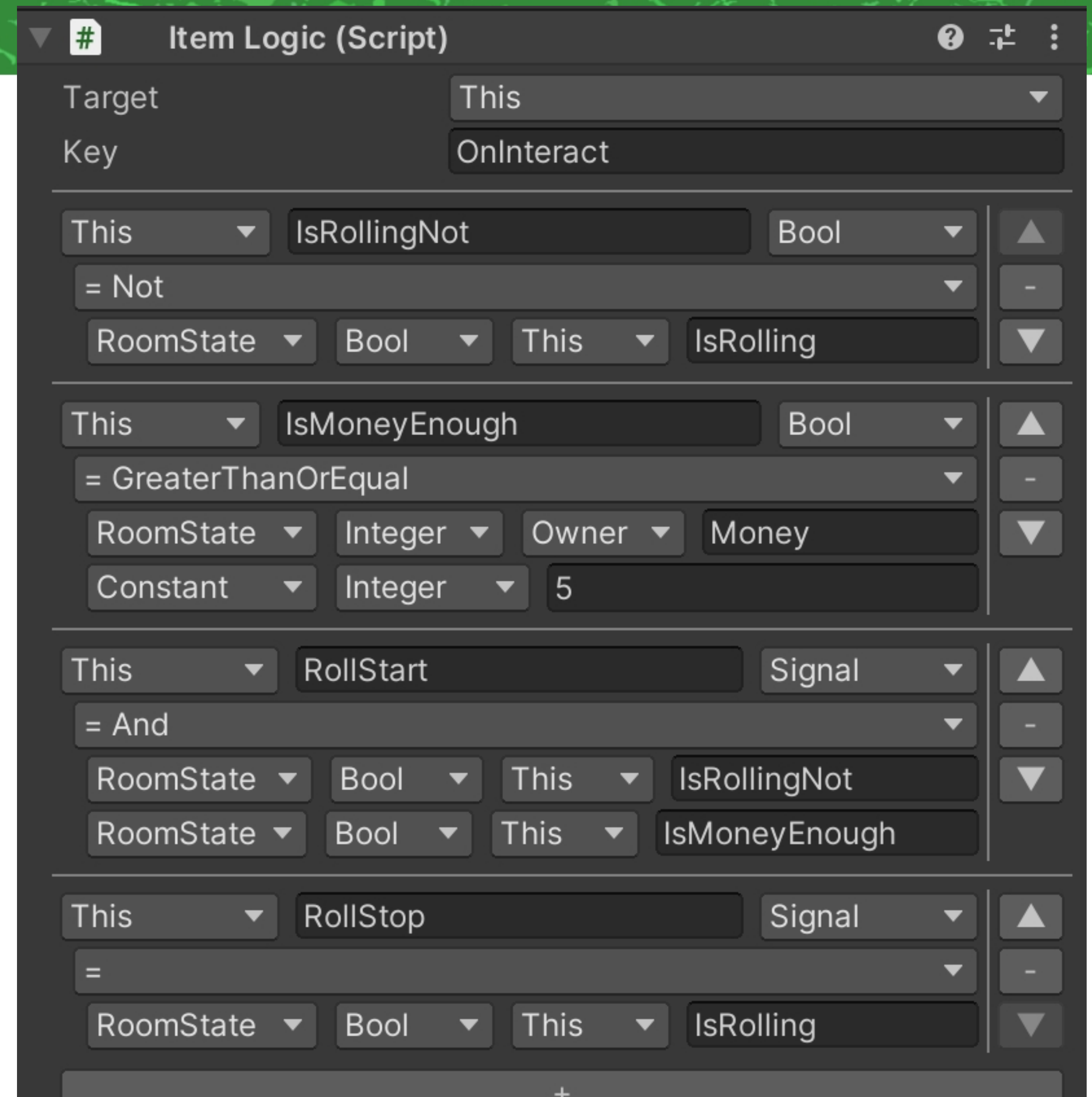


では本体(Machine)を作ります……

すべての起点



InteractItemTriggerから
ItemLogicを呼び出す



「お金があり、しかもまだ
スロット開始してない」

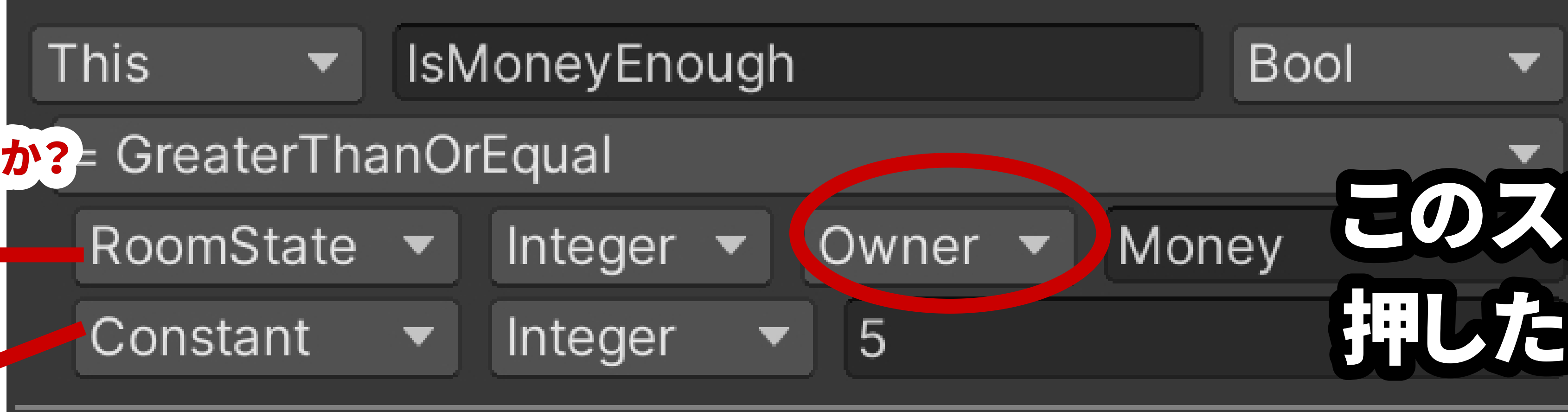
一度個別に
フラグ判定して
And(しかも)で
最終判定

こういうときは
Bool(ON/OFF)の変数を使う

The screenshot shows a logic editor interface with three conditions stacked vertically:

- Condition 1:** Variable: This; Property: IsRollingNot; Type: Bool. Logic: = Not IsRollingはOFFですか? (IsRolling is OFF?). Sub-conditions: RoomState (Bool) This (IsRolling).
- Condition 2:** Variable: This; Property: IsMoneyEnough; Type: Bool. Logic: = GreaterThanOrEqual 押した人の金ありますか? (Money of the person who pressed?). Sub-conditions: RoomState (Integer) Owner (Money) Constant (Integer) 5.
- Condition 3:** Variable: This; Property: RollStart; Type: Signal. Logic: = And その2つが両方OKならRollStart起動! (If both are OK, RollStart starts!). Sub-conditions: RoomState (Bool) This (IsRollingNot) RoomState (Bool) This (IsMoneyEnough).

**Ownerの話は深入りすると大変なので
あまりやりませんが**



○○以上かどうか? = GreaterThanOrEqualTo

変数 — RoomState Integer **Owner** Money

ただの数 — Constant Integer 5

**このスロットを
押した人のお金**

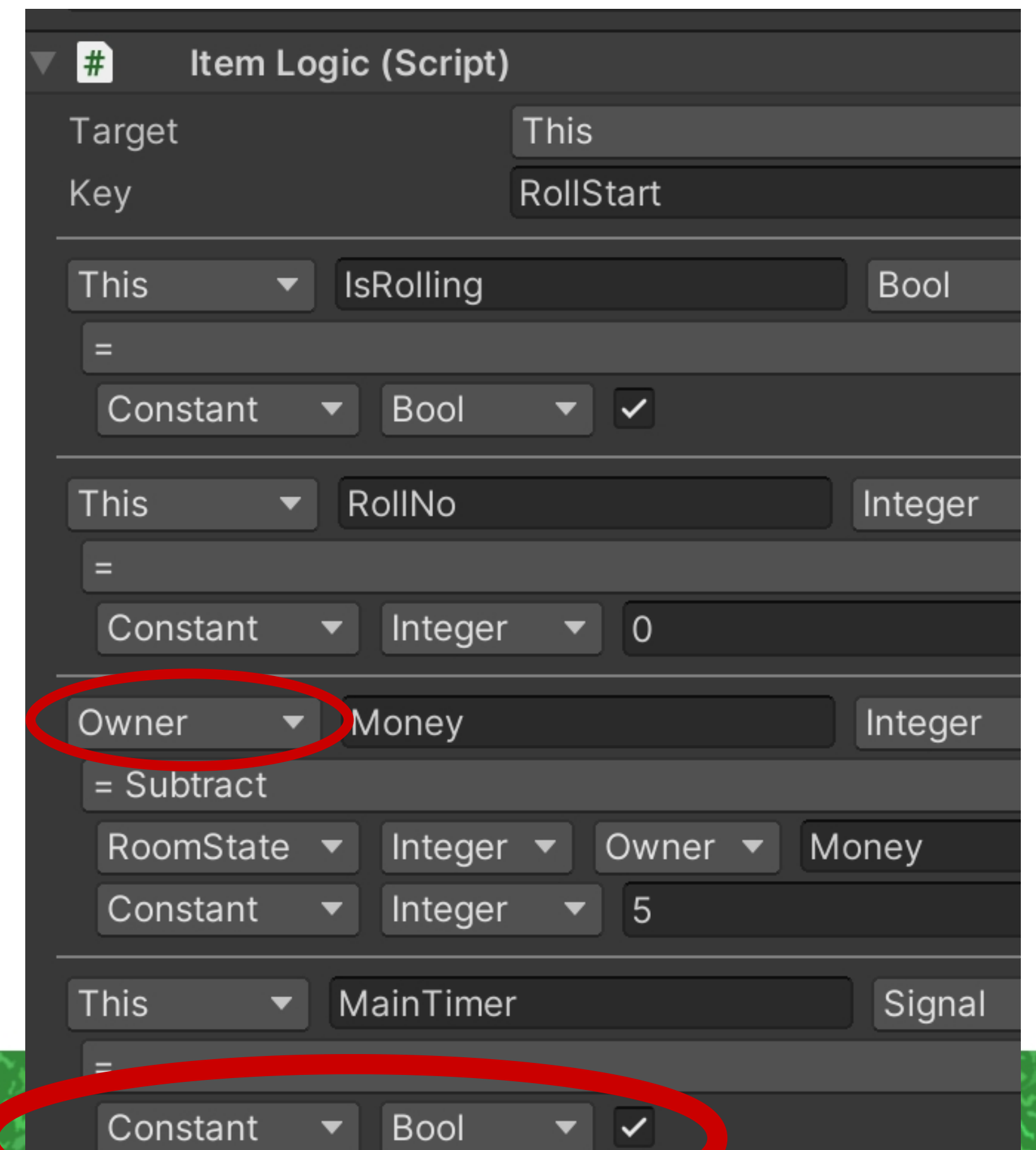
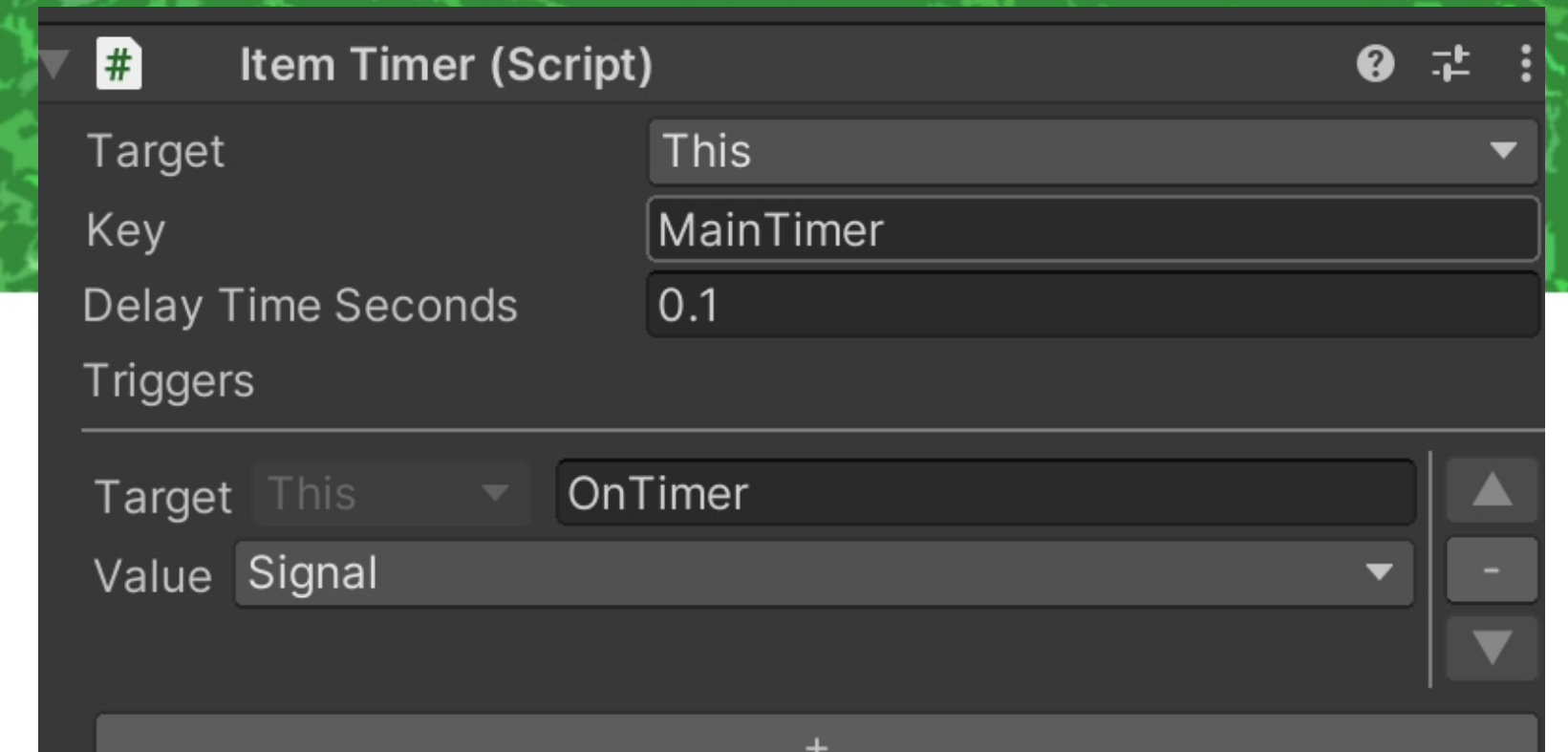
「クリックしたItemのOwnerは、
クリックした人になる」は覚えておこう

ItemTimerは、ループさせるかどうか、後で判定したいから
今回**OnTimer**を呼び出すだけ

IsRolling(開始したか)はON
RollNo(止まった絵の数)は0
Ownerの**Money**を-5
Timerを呼び出す

今回はOwnerに
残金を保存

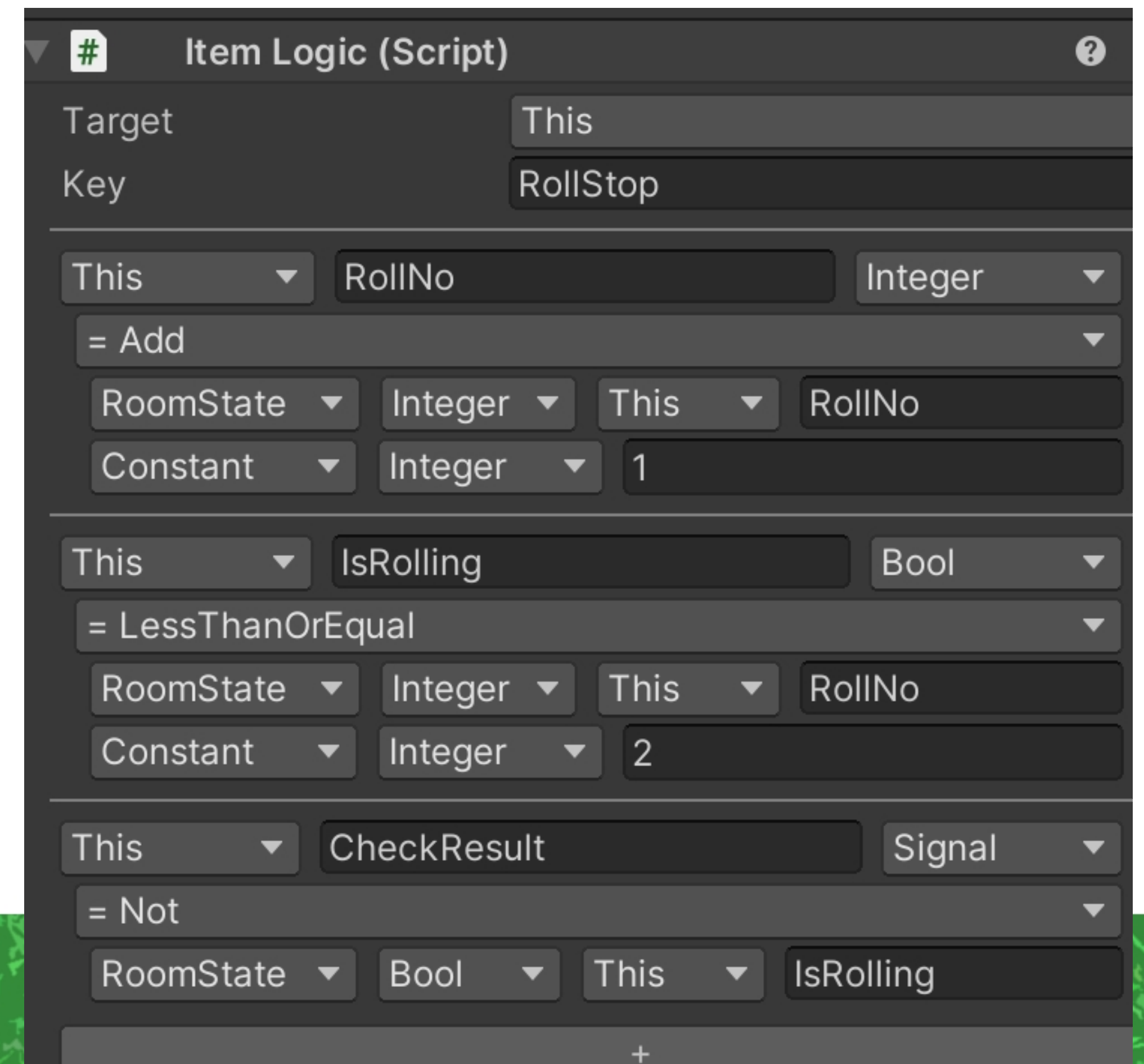
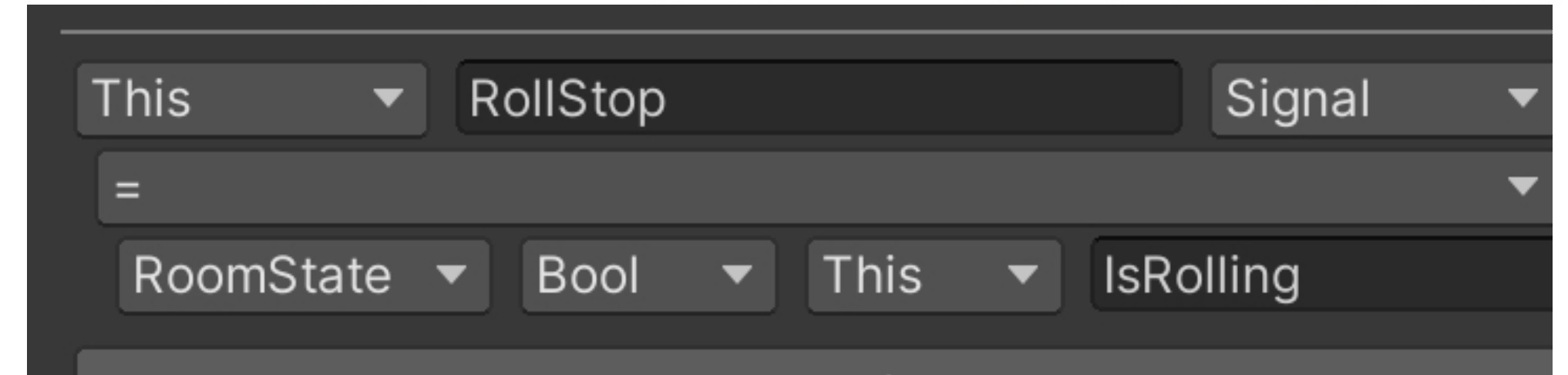
常に**Signal**送りたいときはこう!



OnInteractの後半 →

すでにスロット開始している場合は
RollStopのItemLogicが呼ばれる

- **RollNo**という変数+1
- まだ2以下なら**IsRolling:ON**
2を越えた(3以上)ならOFFにする
- で、**IsRolling**がOFFなら
CheckResult起動!



OnTimerのLogic

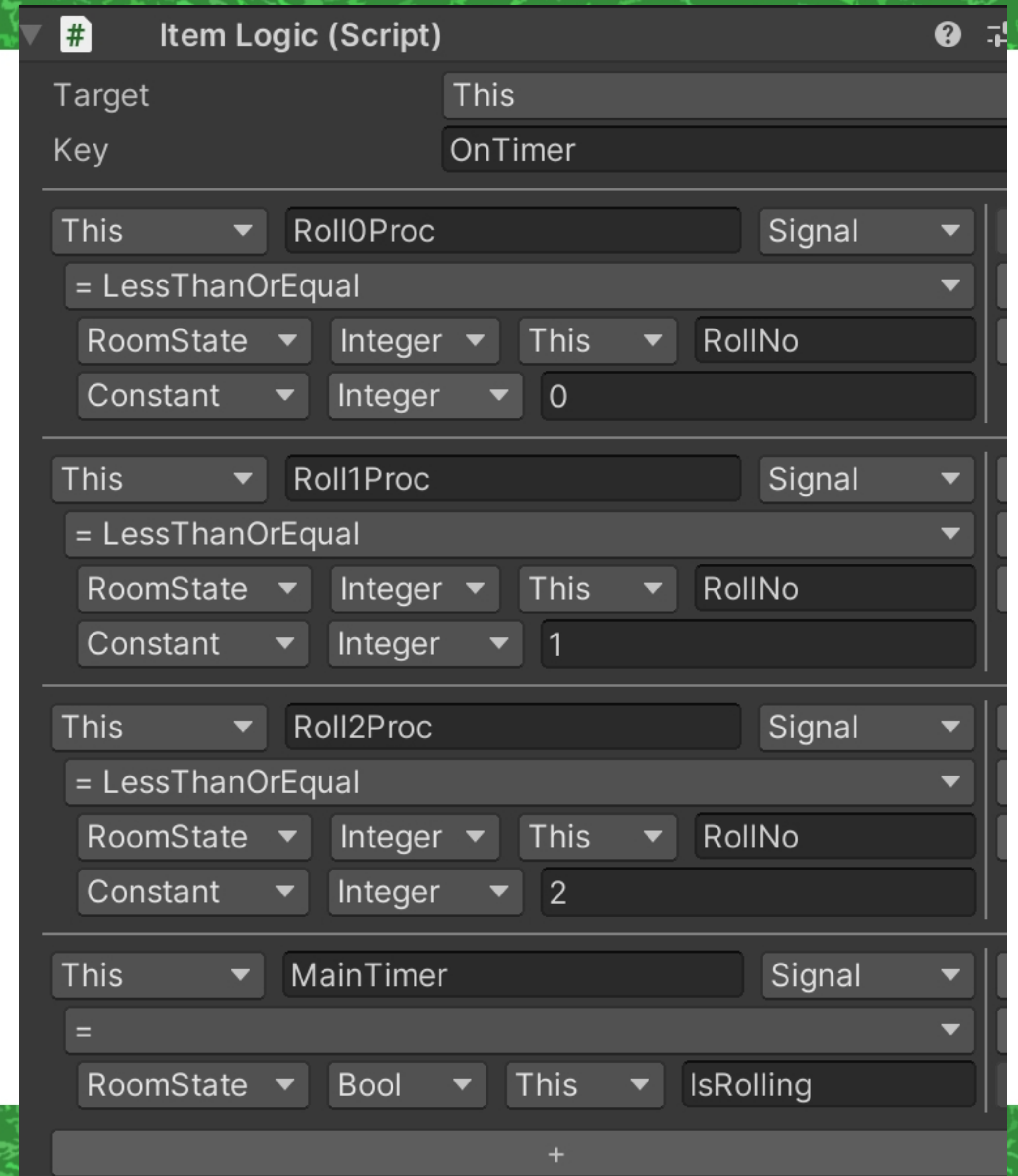
RollNoが0なら、絵柄全部変更

1なら右の2つだけ変更

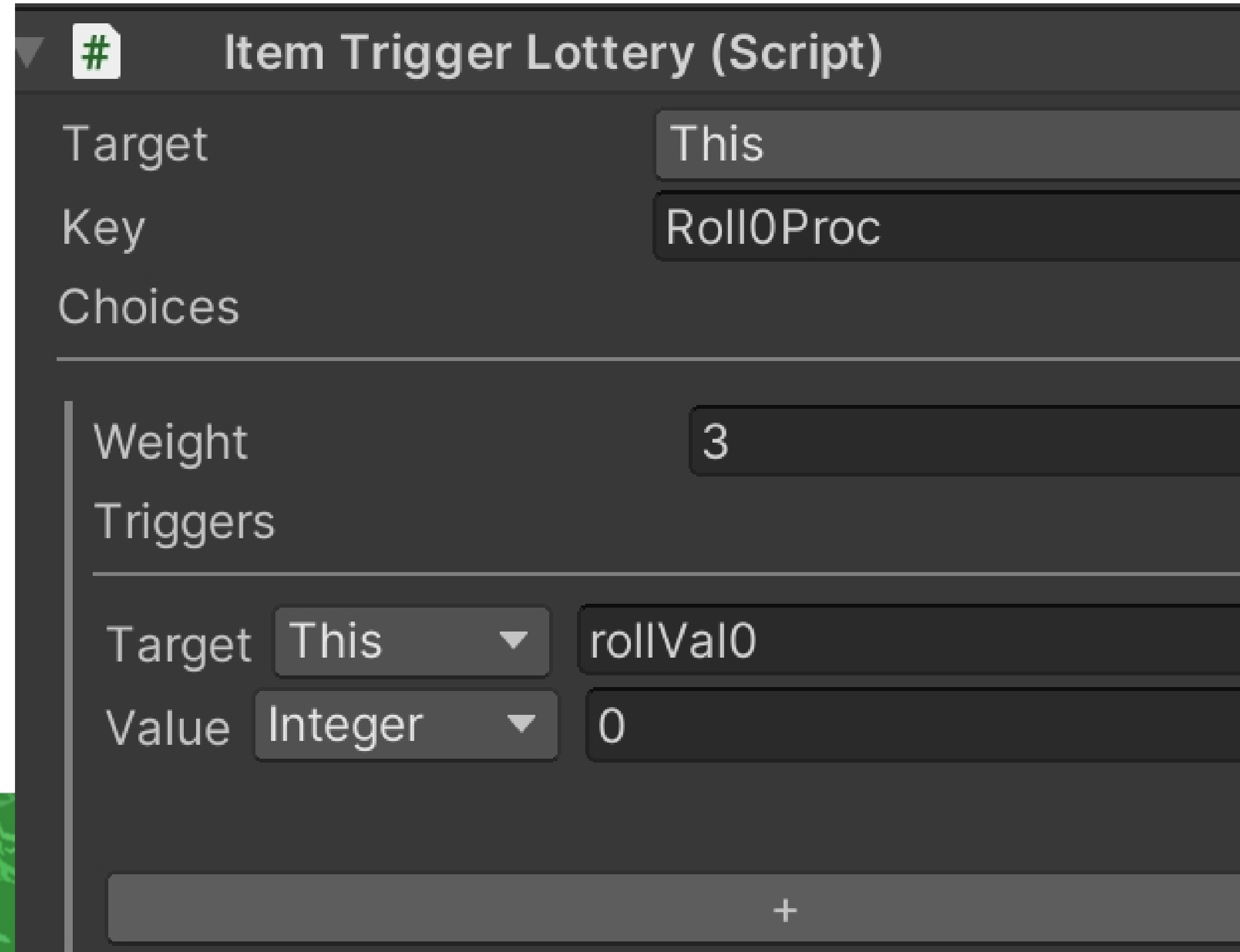
2なら一番右だけ

そして**IsRolling**がONの時だけ

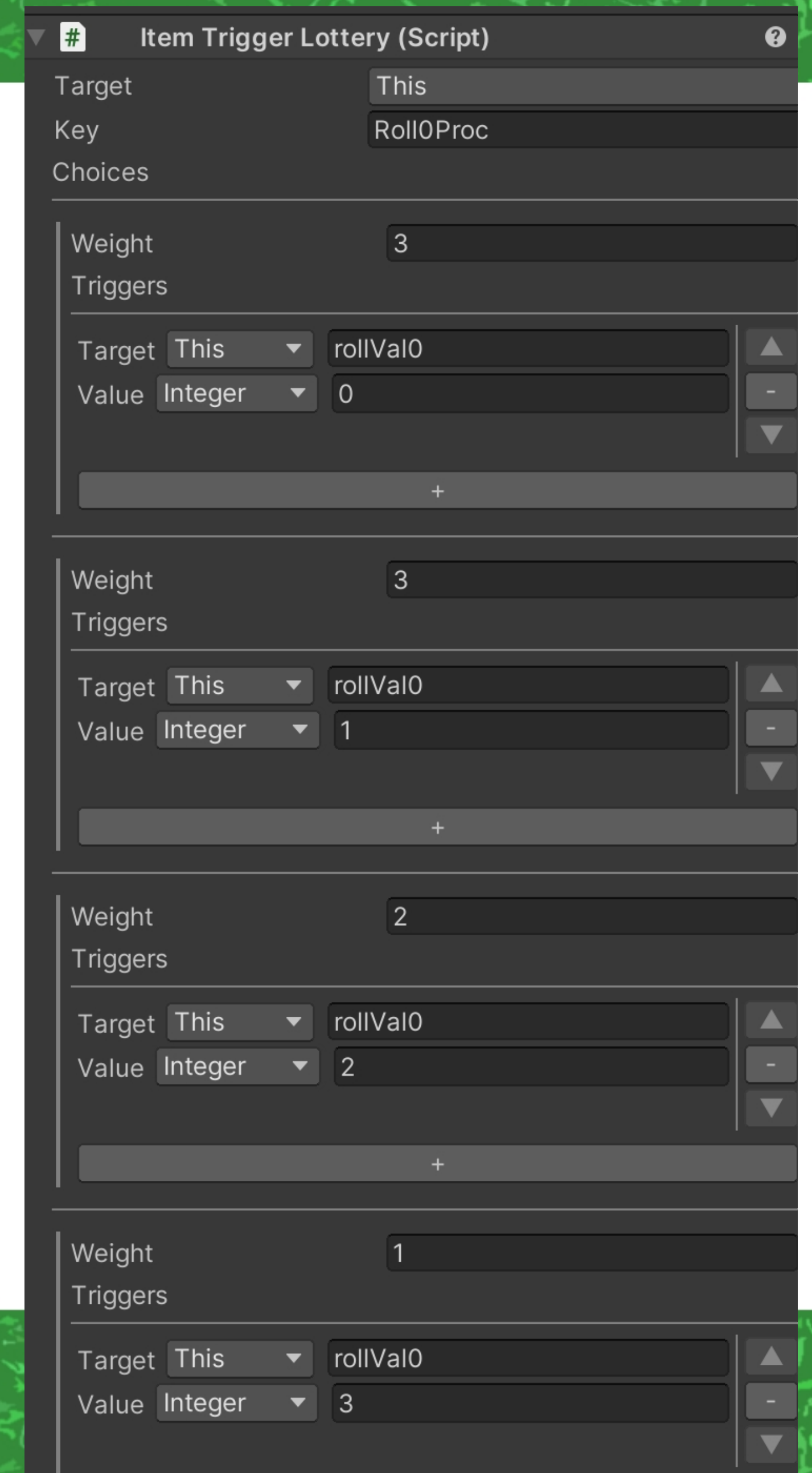
次のタイマーを起動

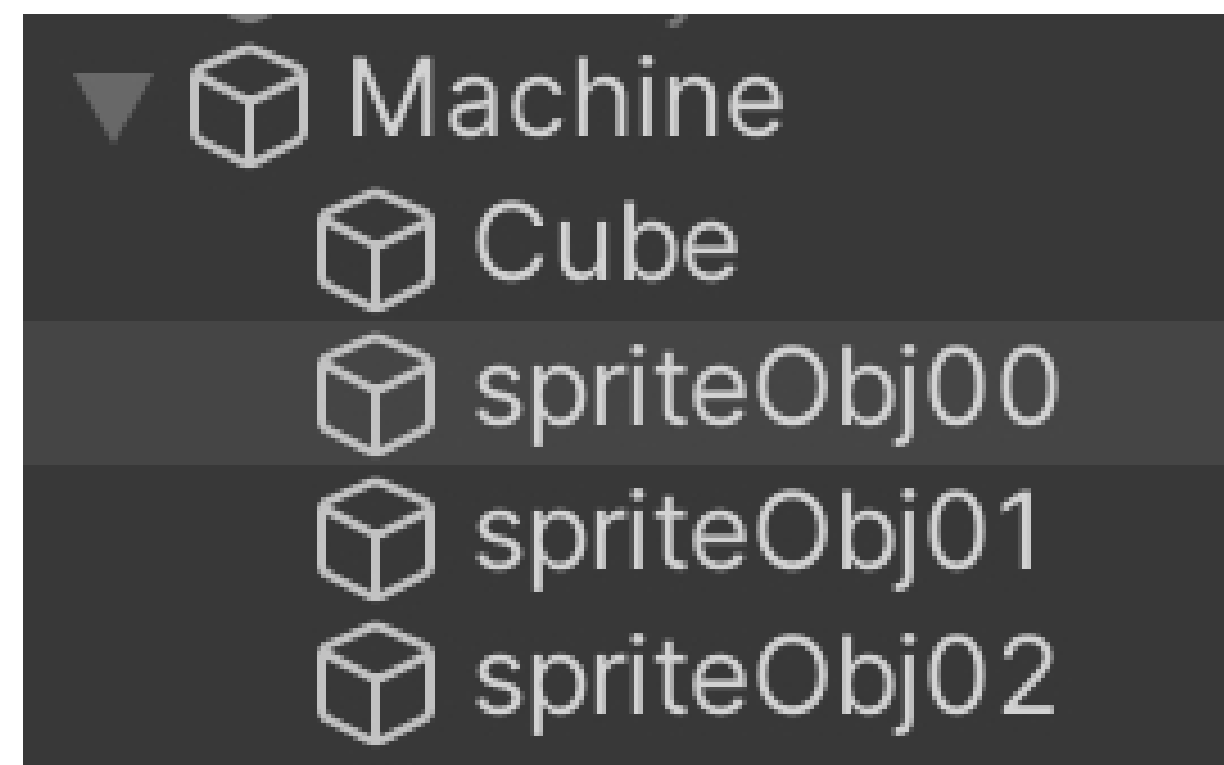


絵柄変更 (**Roll0Proc**とか)は **ItemTriggerLottery** (くじびき)

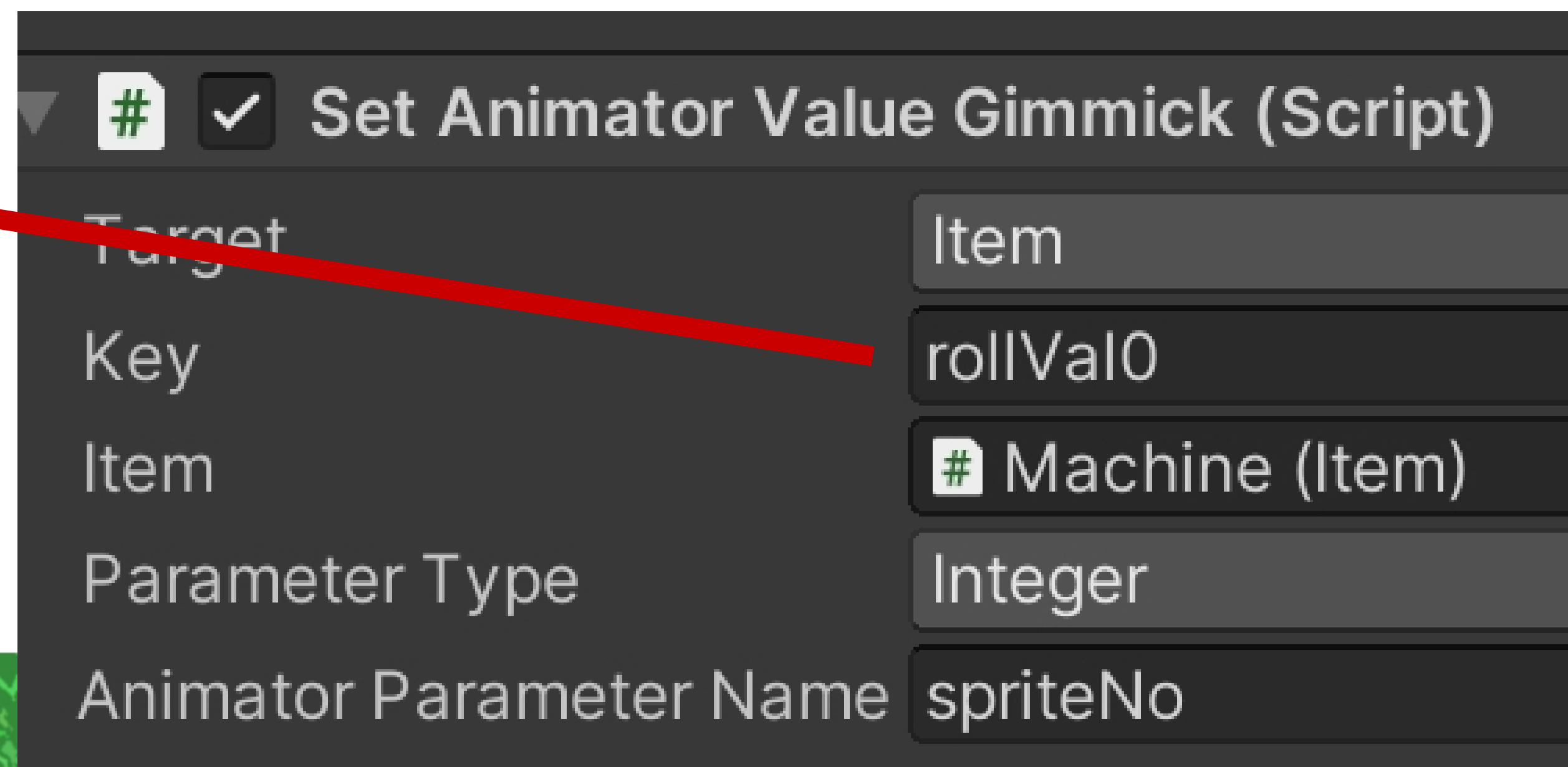
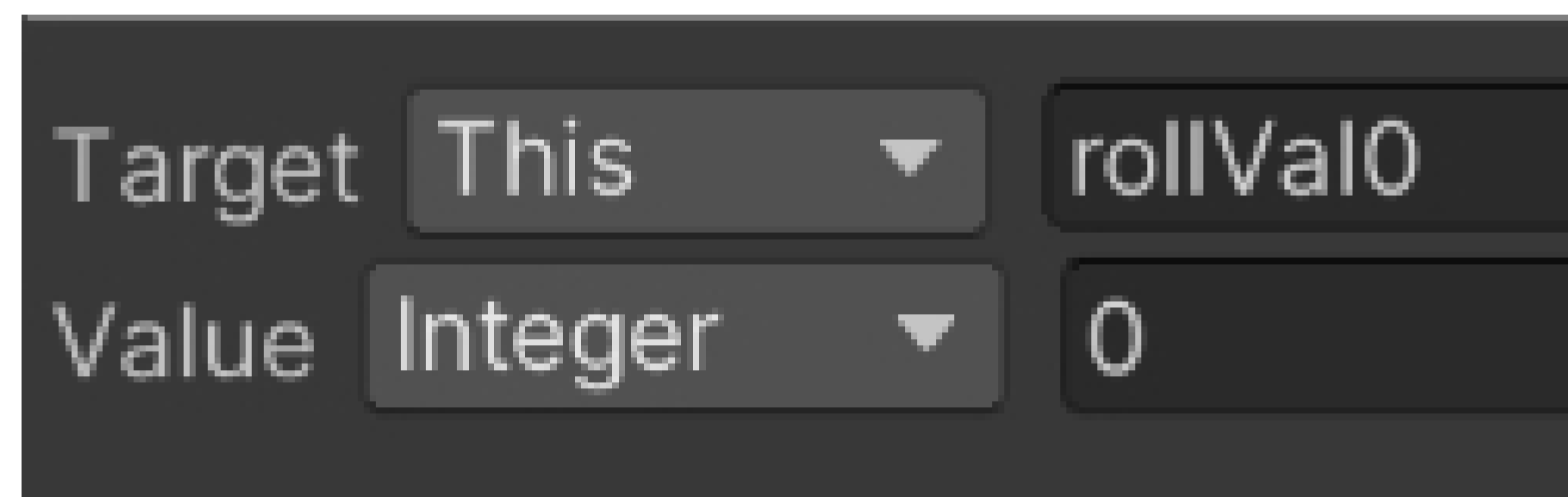


←一部拡大





Lotteryから数字が入ると、
各spriteObjにつけてた
SetAnimatorValueGimmickが
反応して絵柄が変わる!



他のLogicとかGimmick起動するのは
基本**Signal**だが……



SetAnimatorValueGimmickみたいに
Integerとかで起動するのもあるので
中級者になってきたら覚えておこう

ちなみにLotteryでは
weight(重み)を
それぞれ変えていて
絵柄の出やすさが
ビミョーに違う

Weight 3

Triggers

Target This rollVal0

Value Integer 1

+

Weight 2

Triggers

Target This rollVal0

Value Integer 2

+

Weight 1



全部止まったら、役判定!

「2つ同じのがある」か
「3つ同じのがある」で
お金が入るルール

左から「A」「B」「C」とする

→ **A=B、A=C、B=Cと**

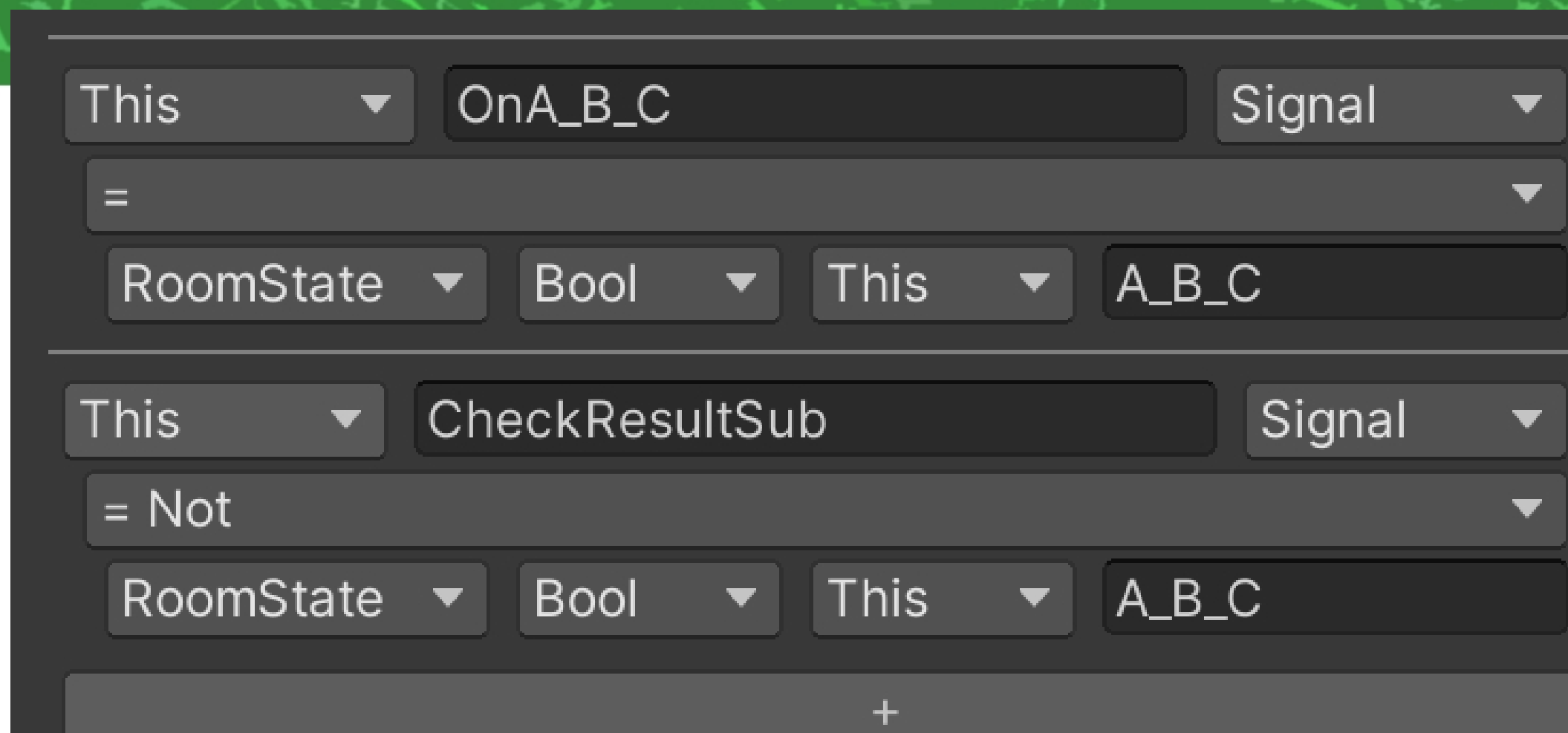
A=B=Cを判定・保存

The screenshot shows the 'Item Logic (Script)' interface with three logic rules defined:

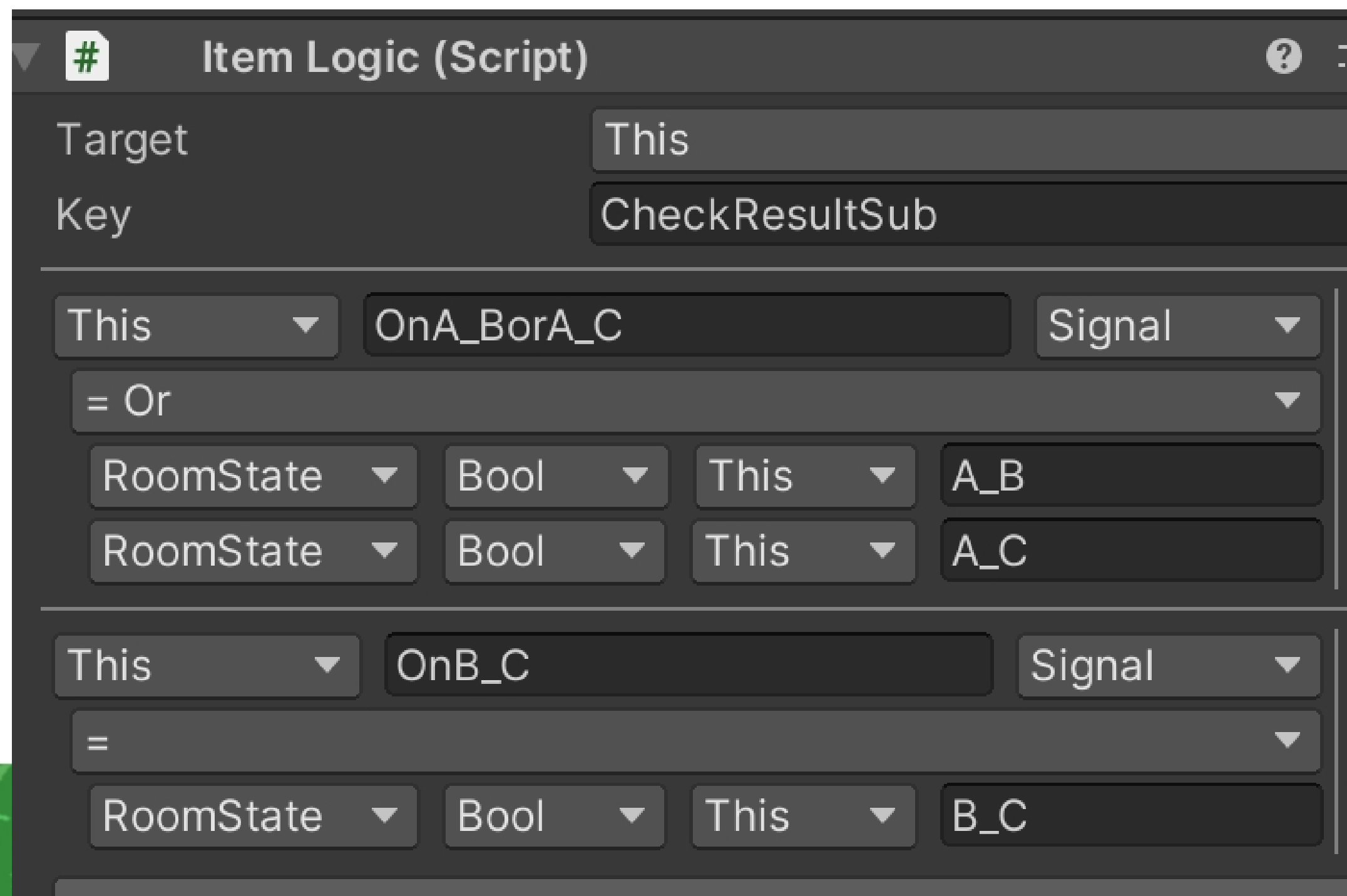
- Rule 1 (A_B):** Target: This, Key: CheckResult. Logic: This (Bool) = Equals (RoomState Integer, This rollVal0, RoomState Integer, This rollVal1).
- Rule 2 (A_C):** Target: This, Key: CheckResult. Logic: This (Bool) = Equals (RoomState Integer, This rollVal0, RoomState Integer, This rollVal2).
- Rule 3 (B_C):** Target: This, Key: CheckResult. Logic: This (Bool) = Equals (RoomState Integer, This rollVal1, RoomState Integer, This rollVal2).

Below these, a fourth rule is partially visible:

- Rule 4 (A_B_C):** Target: This, Key: CheckResult. Logic: This (Bool) = And (RoomState Integer, This A_B, RoomState Integer, This A_C).



A=B=Cなら「3つそろった」処理へ
そうじゃないなら
CheckResultSubに飛ぶ



A=B、またはA=C

→2つそろった。Aの絵柄を見る

B=C

→2つそろった。Bの絵柄を見る

Item Logic (Script)

Target This

Key OnA_B_C

Owner ▼ GotVal50 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ rollVal0

Constant ▼ Integer ▼ 0

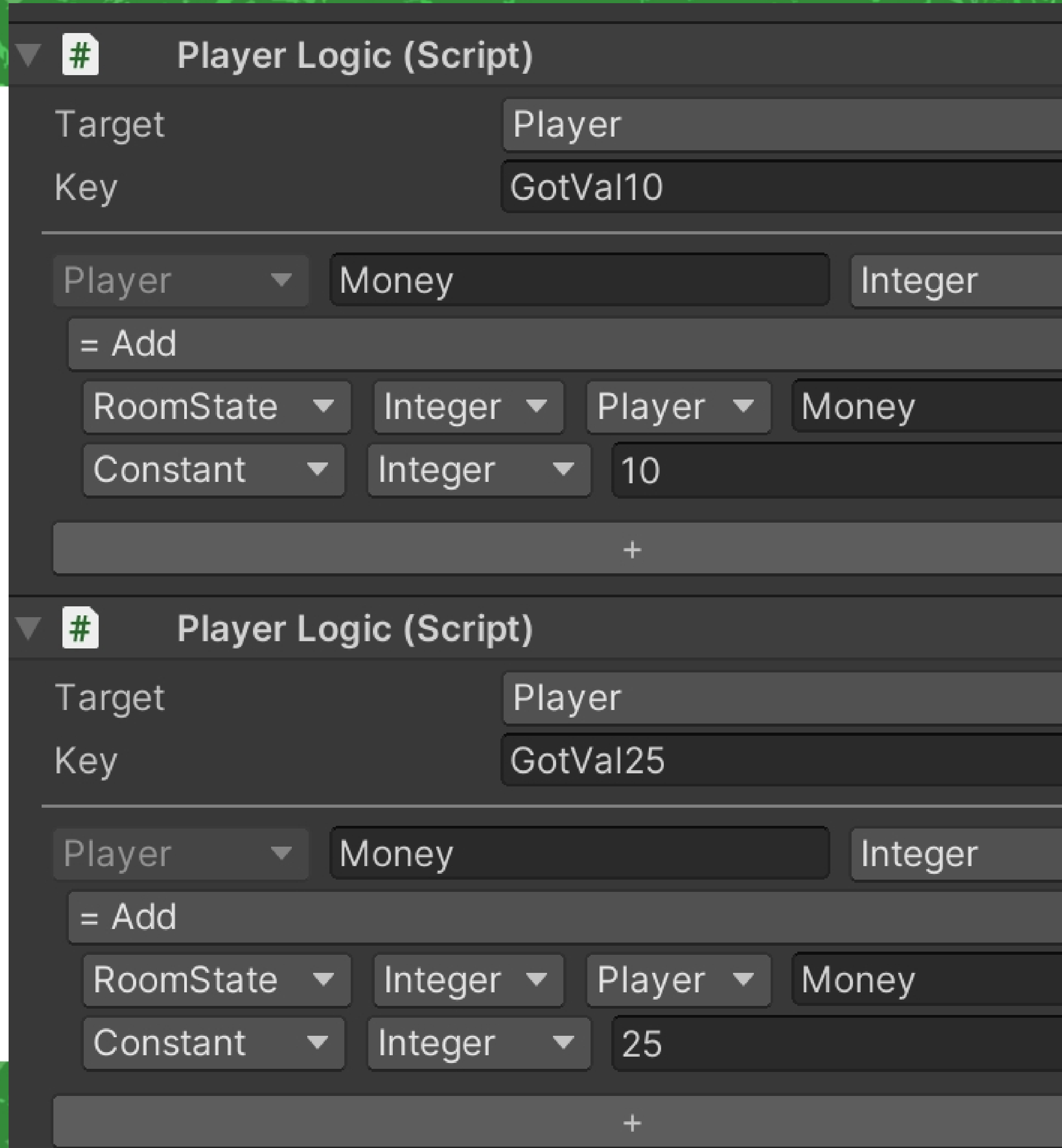
Owner ▼ GotVal80 Signal

= Equals

RoomState ▼ Integer ▼ This ▼ rollVal0

Constant ▼ Integer ▼ 1

あとは絵柄によって
ゲットするお金の
メッセージを変えて
**Ownerの
PlayerLogicを
呼び出す。**



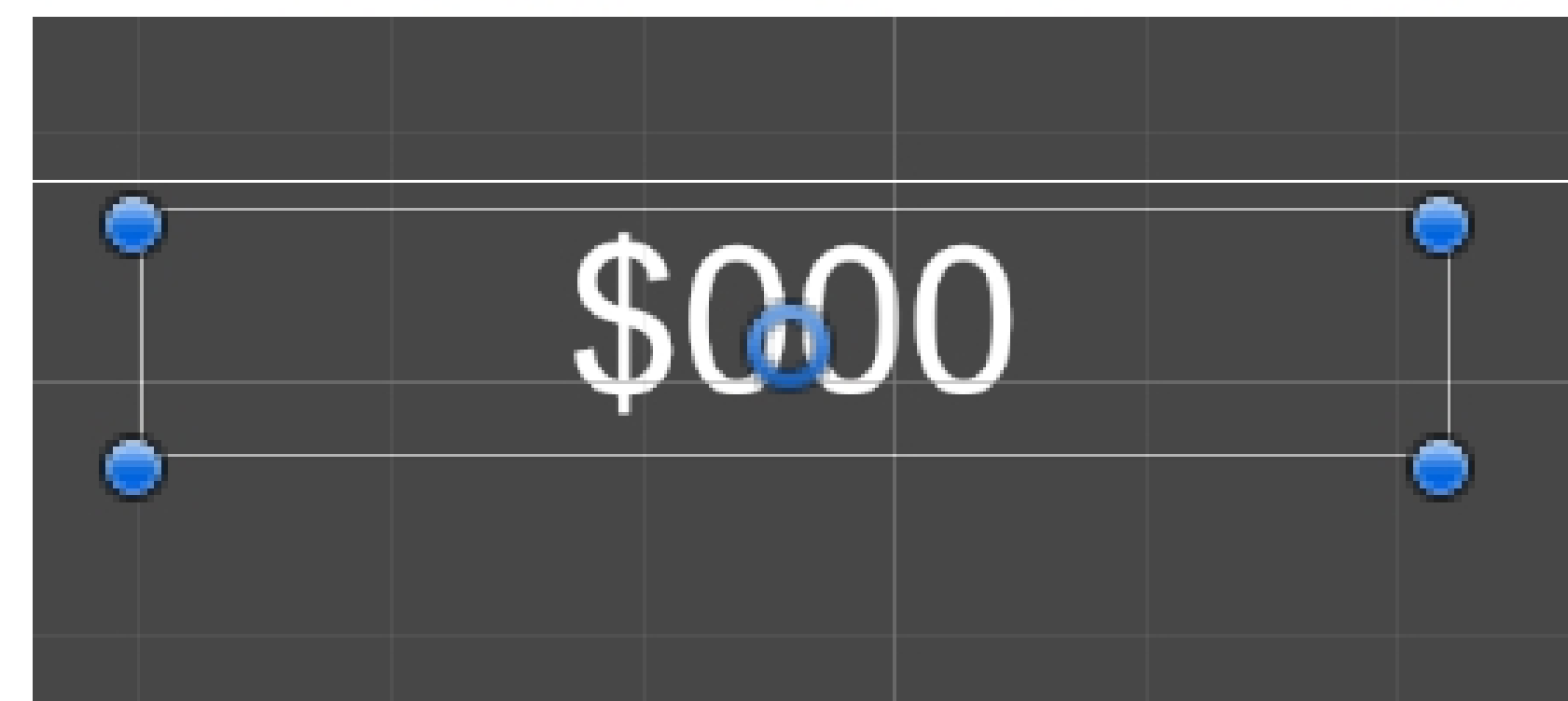
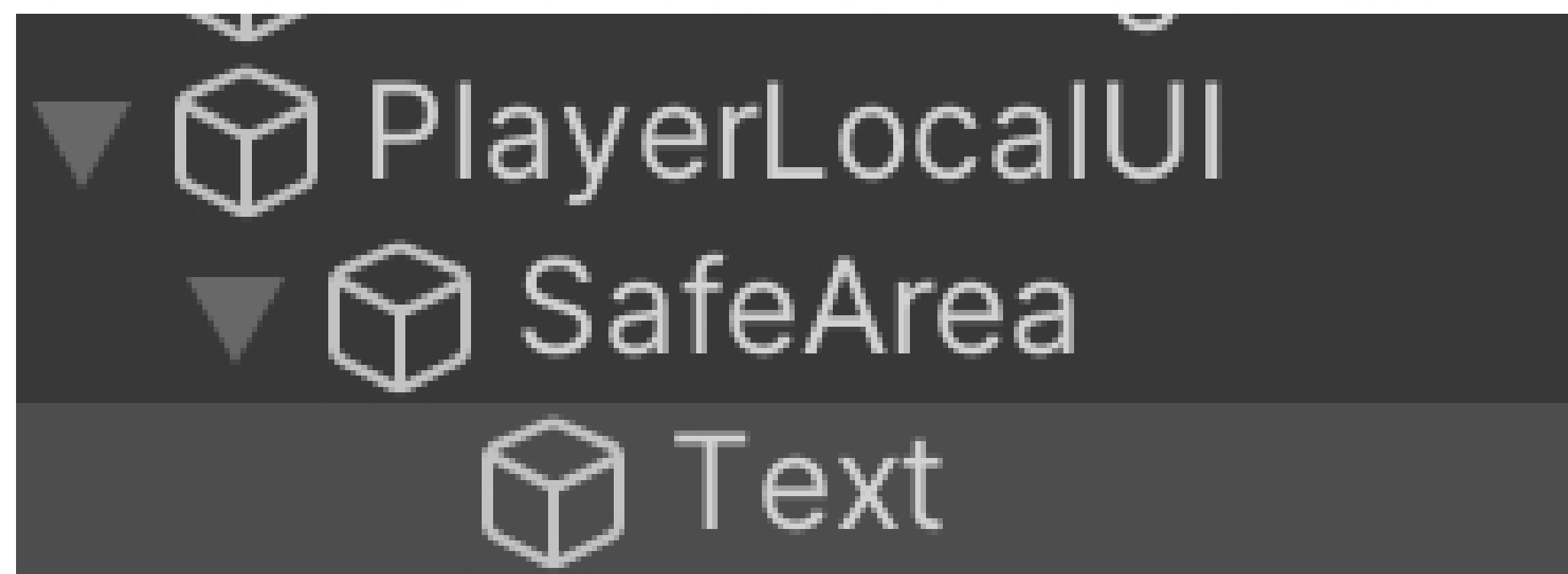
**EmptyObjectに
PlayerGotManagerとか
適当な名前をつけて
お金を増やすPlayerLogicを
ひたすら付けていく**

ちなみにUnityのコンポーネントは
コピペができるんで
こういうときは使うと便利

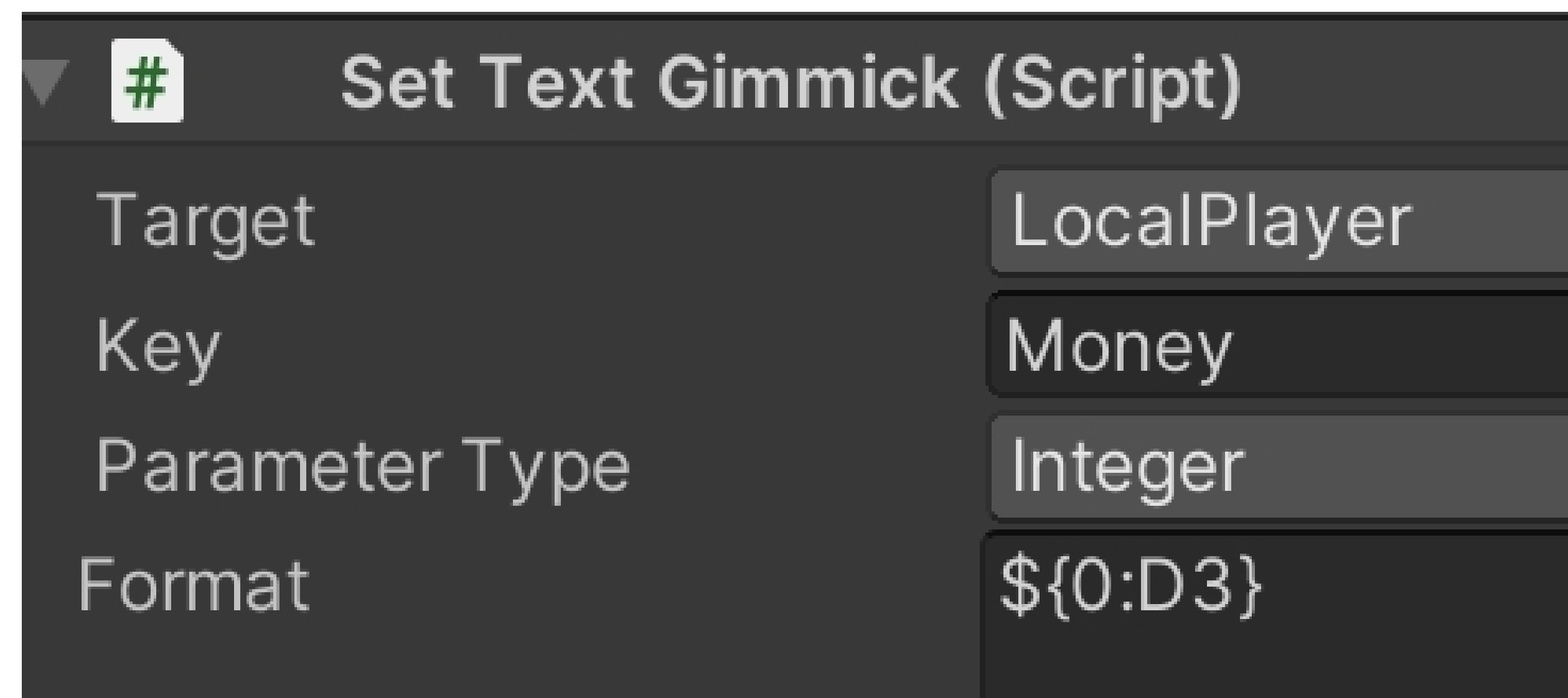
Logic地獄.....

あとなんかLogic多いItemはUnityでの編集が重い。

なお、回避方法はない



あと、お金は
PlayerLocalUIと
SetTextGimmickで
表示しときましょう



前に
やった

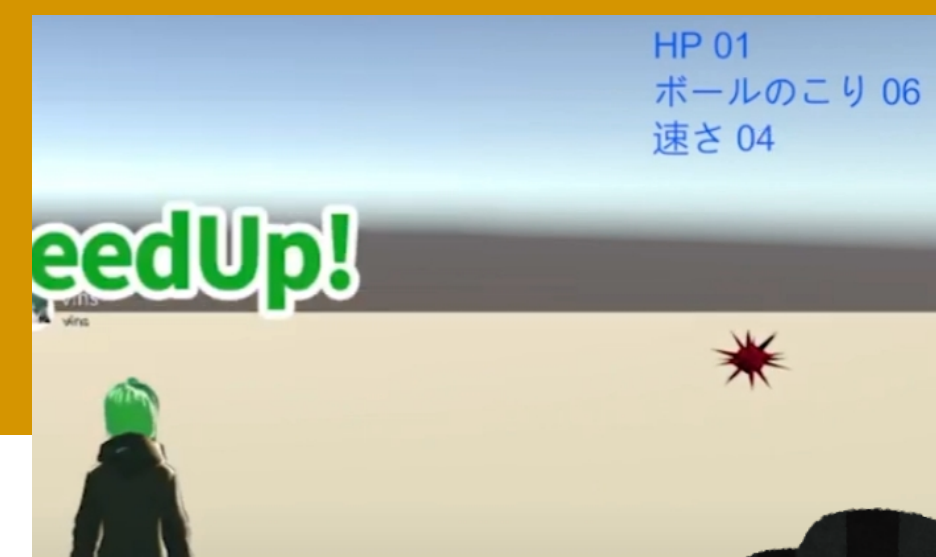
Logicのよくあるパターン 剣で戦うRPGイメージで

ここの掘り下げは**第3回ゆるゆる勉強会**の
PDFと動画で
(イベントページにリンクを張っています)



前に
やった

Logic使ったゲームワールドを 作ろう パターン編2



ここの掘り下げはvinsのZennの記事で
(イベントページにリンクを張っています)



まあ、大変ではあるものの
Logicを使うと
幅がすごく広がりますよ





一步一步。

ありがとうございました!