BP設計力を培おう！

pate（ぱて）

もくじ

1. BP設計力とは？

1. BP設計力とは？

BP設計力とは？

BP設計力 ＝ 拡張性・可読性・保守性に考慮してBlueprintを設計し実装する能力

BP設計力はSOLID原則のことだったのか！？

BP設計力 ≒ SOLID原則

BP設計力・・・とは・・・？

BP設計力 ＝ SOLID原則を適切に
取り入れつつ
BPを実装する力

~~BP設計力を培おう！~~

一部

SOLID原則をBP実装例から解説してみる

pate（ぱて）

真のもくじ

1. SOLID原則とは？
2. SOLID原則のメリデメ
3. SOLID原則のBP実装例を1つだけ紹介（時間ねえ）
4. まとめ
5. 告知

1. SOLID原則とは？

SOLID原則とは？

ソフトウェア設計における5つの基本原則の頭文字をとったもの

拡張性、可読性、保守性の高いシステムを構築するための基本指針

1. Single Responsibility Principle（単一責任の原則）
2. Open/Closed Principle（オープン・クローズドの原則）
3. Liskov Substitution Principle（リスコフの置換原則）
4. Interface Segregation Principle（インターフェース分離の原則）
5. Dependency Inversion Principle（依存関係逆転の原則）

ふーん、いい感じの設計思想ってことね。

ならこれから全部取り入れてみるか。

SOLID原則とは？

やりすぎ、ダメ！！！

SOLID原則に固執する、ダメ！！！

2. SOLID原則のメリデメ

SOLID原則のメリデメ

SOLID原則を守るメリット

- 拡張性の向上　→　機能追加、再利用しやすい
- 可読性の向上　→　理解しやすい
- 保守性の向上　→　変更、修正しやすい

SOLID原則を守るデメリット

- 複雑になる
- 初期開発速度の低下
- オーバーエンジニアリング（必要以上に複雑にする、不要な機能を作る）
- 学習コストの増加

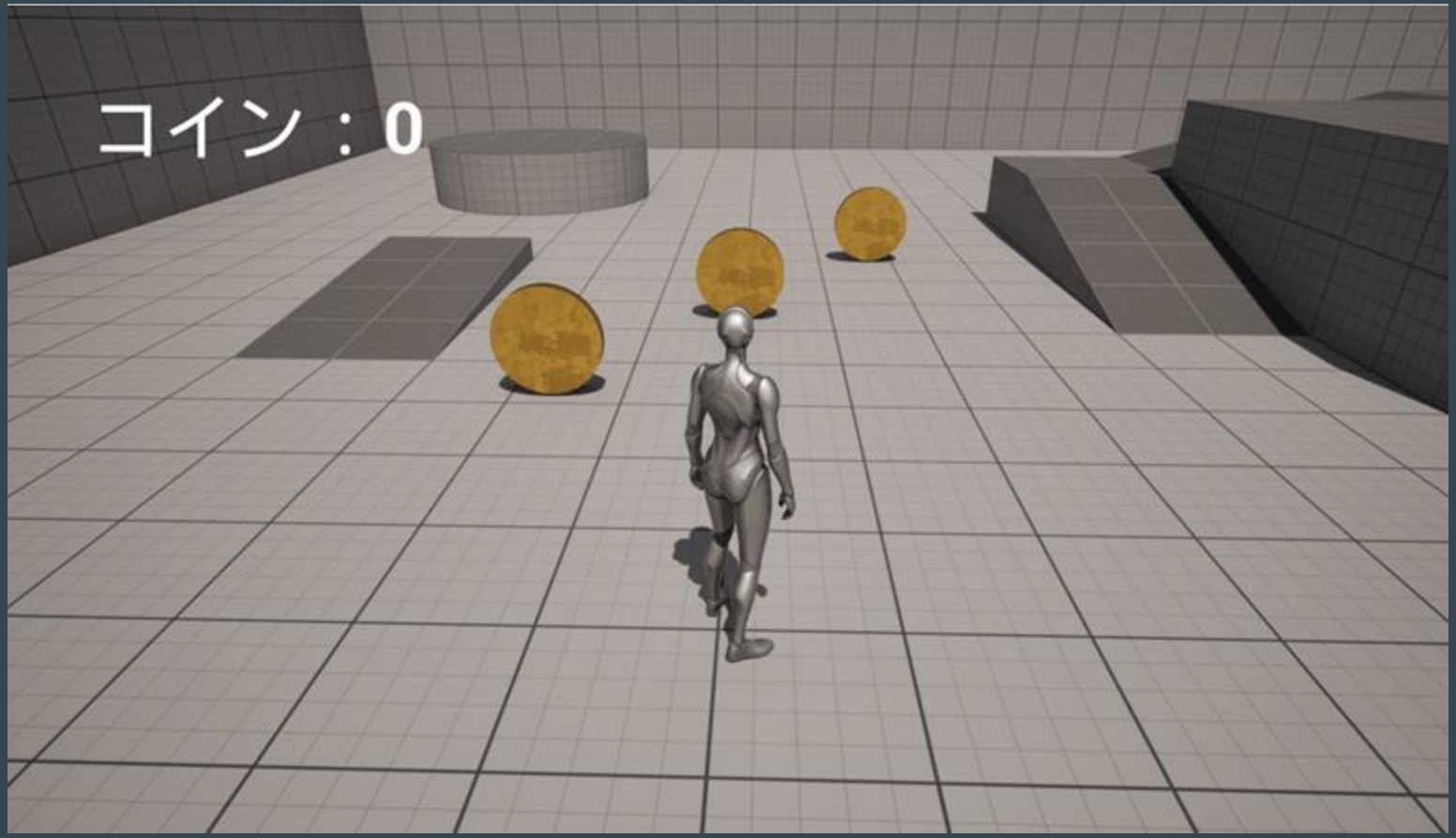
3. SOLID原則のBP実装例を1つだけ紹介

Single Responsibility Principle（単一責任の原則）

モジュール、クラスまたは関数は、単一の機能について責任
を持ち、その機能をカプセル化するべきであるという原則

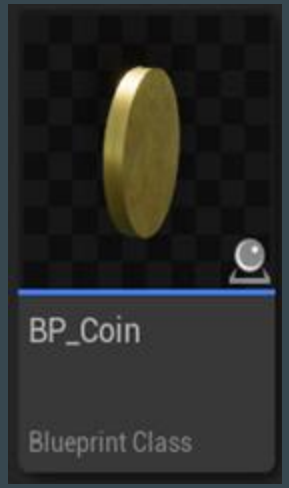
1つのBPになんでもかんでも機能実装するんじゃねえ！

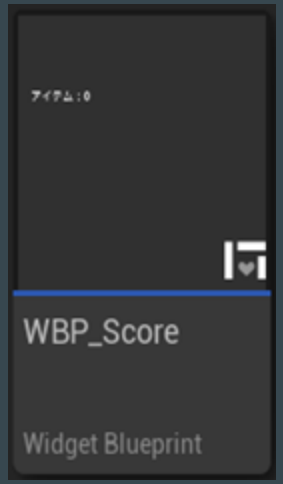
Single Responsibility Principle（単一責任の原則）



Single Responsibility Principle（単一責任の原則）

登場人物から設計を考える

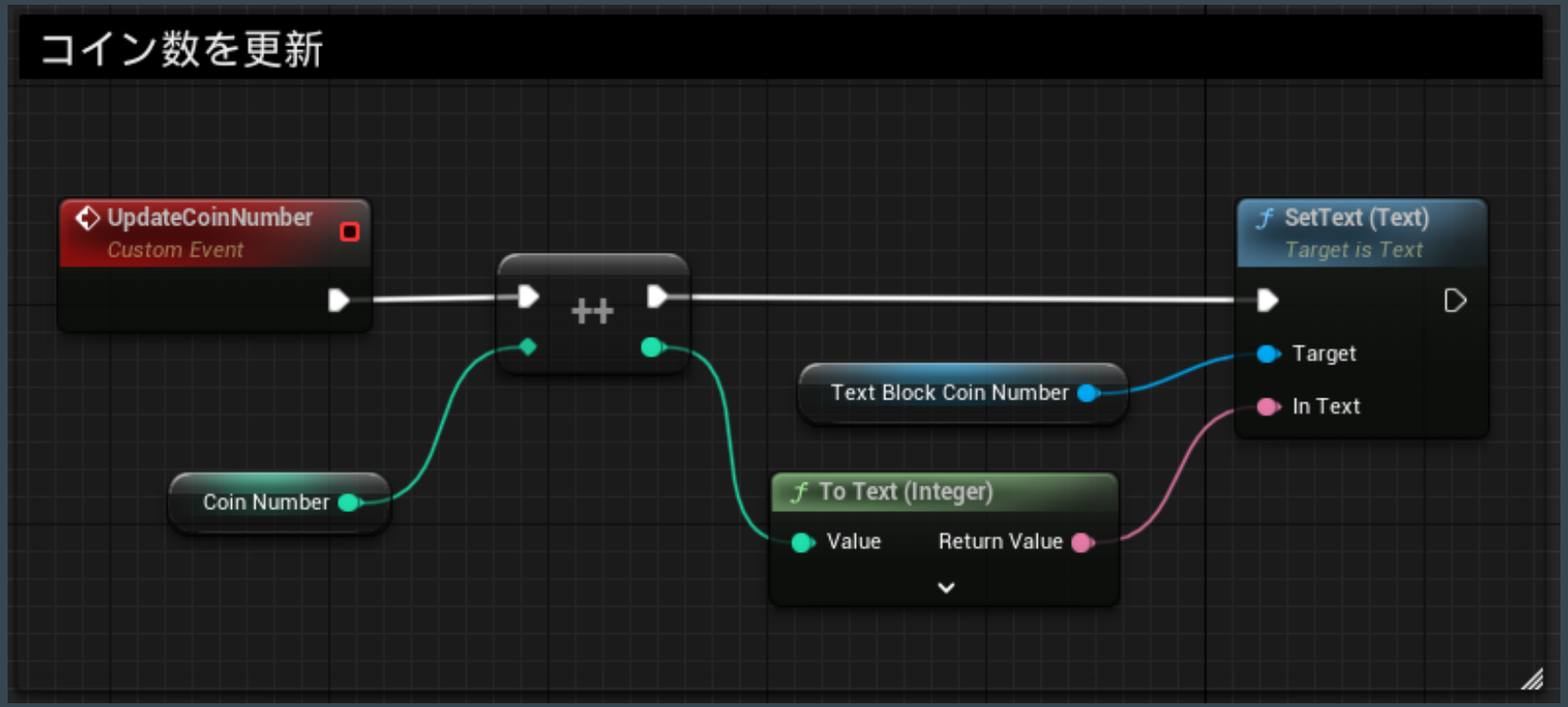




こいつをどこに持たせるか

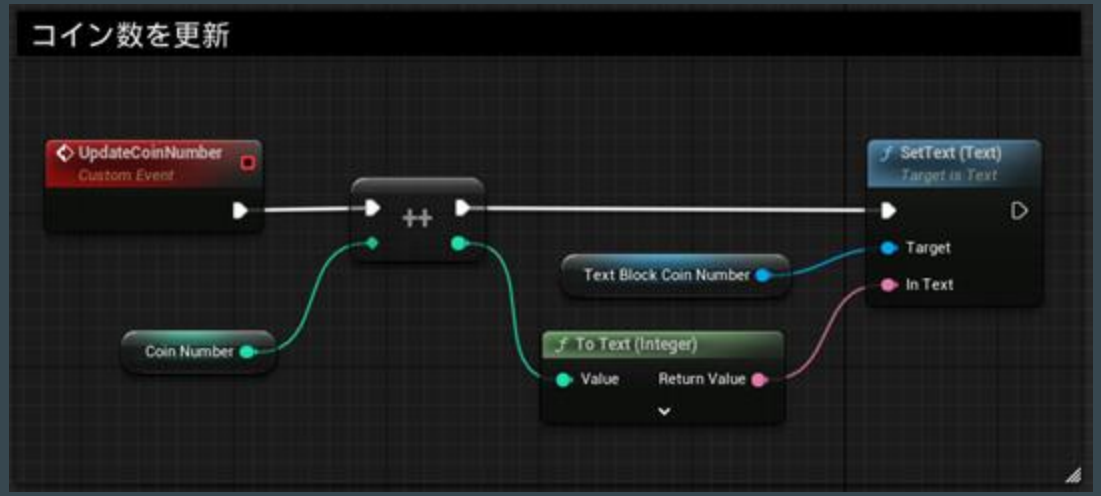
単一責任の原則を破っている例

Widgetに持たせるとどうなるか



単一責任の原則を破っている例

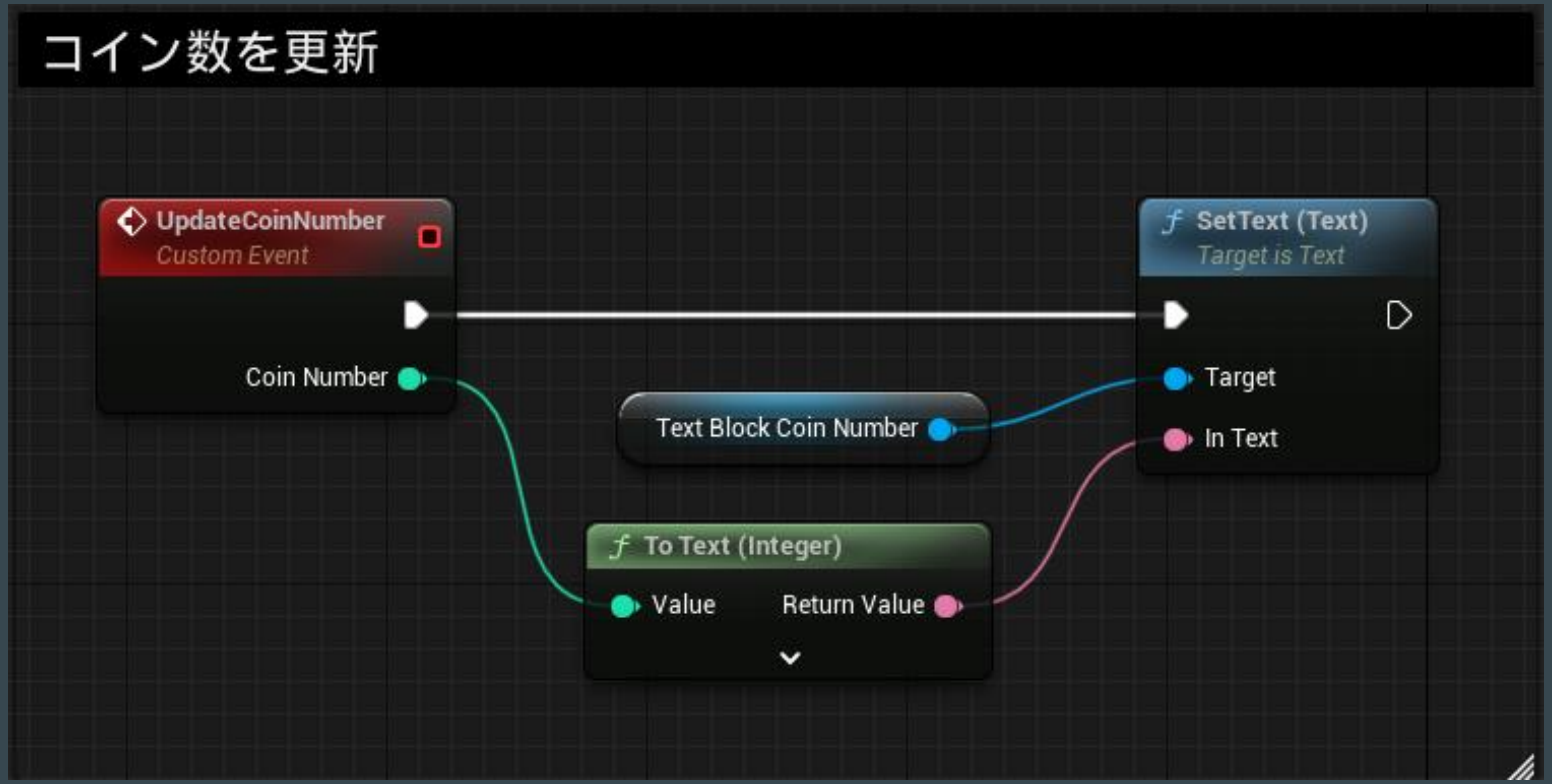
Widgetに持たせるとどうなるか



- Text変数に値を格納する役割
- CoinNumberの値を更新する役割
- Coin獲得枚数を定義する役割

受け取った値をText変数に格納する役割

単一責任の原則を守っている例



まとめ

30分枠にしてもっと紹介したらよかった・・・

興味があればSOLID原則を調べてみてください

告知

富山でUEイベントやります！！！