第12章
メソッド

この章の目的

◆メソッドの設計について振り返りも含めて学ぶ

内容

内容とキーワード

◆今まで学んだこと
◆コマンド・クエリ分離
◆引数
◆戻り値

今まで学んだこと

今まで学んだこと

◆必ず自身のクラスのインスタンスを使用すること
- ■低凝集にならないようにする

◆不変をベースに予期せぬ動作を防ぐこと
- ■不変による堅牢性を保つ

◆尋ねるな、命じろ
- ■メソッドは基本的に呼び出される側が処理をし、呼び出す側は呼び出すだけにする

コマンド・クエリ分離

コマンド・クエリ分離

コマンド・クエリ分離とは、メソッドがクエリもしくはコマンドどちらか一方だけを行うように設計すること。

◆以下は状態の取得と変更を同時に行っている

```
int gainAndGetPoint() {
    point += 10
    return point;
}
```

状態の変更

状態の取得

◆状態の変更と取得を同時に行うと、混乱しやすい上に、利用しにくいメソッドになってしまう

メソッド種別	説明
コマンド	状態を変更する
クエリ	状態を返す
モディファイア	コマンドとクエリを同時に行う

コマンド・クエリ分離

コマンド・クエリ分離とは、メソッドがクエリもしくはコマンドどちらか一方だけを行うように設計すること。

◆コマンドとクエリを分けて設計する

■非常にシンプルな作りになる

```
void gainPoint() {
    point += 10;
}

void getPoint() {
    return point;
}
```

状態の変更

状態の取得

引数

引数

- ◆引数は不変にすること
  - ■意図しない変更を避ける
- ◆フラグ引数は使わない
  - ■ストラテジパターンを使おう
- ◆nullを渡さない
  - ■null前提設計は余計なチェックが増える
- ◆出力引数は使わない
  - ■低凝集に陥ってしまう
- ◆引数は可能な限り少なく
  - ■引数が多いと処理が複雑になりがち

戻り値

戻り値

◆「型」を使って戻り値の意図を表明する
- ■プリミティブ型で返すと、意図が分からなくなる
- ■値オブジェクトを上手く使って意図を表明する

◆nullを返さないこと
- ■引数にnullを渡さないのと同様です

◆エラーは戻り値で返さず例外をスローすること
- ■例えば-1などをエラー値として扱うことがあるが…
- ■うっかりエラー処理を実装し忘れると正常値として扱われてしまう。（ダブルミーニング）

この章で学んだこと

◆メソッドの設計について学んだ
◆クラスと共にメソッドも正しく設計しよう