



テストコードの 注入から始める レガシーコードの リファクタリング

風間 裕也

The background is a photograph of a child's game board, tinted green. A blue pushpin is stuck into the board. To the left of the pushpin are two white dice. Below the pushpin, the word "START" is printed in large, bold, black letters. To the left of "START", there are several black arrows pointing in a circular path. The overall image has a soft, slightly blurred quality.

はじめに

はじめに

- 本発表のゴール
 - テストコードを少しずつ注入して、レガシーコードに立ち向かう方法を知る
- 本発表の対象聴講者
 - プログラミングをある程度経験している
 - Javaを少しは知っている
 - TDD（テスト駆動開発）を少しは知っている
 - 「レガシーコード」という言葉を聞いたことがある
 - 特に、プログラムが全く読めない、分からないという方はついてこれない可能性があります。

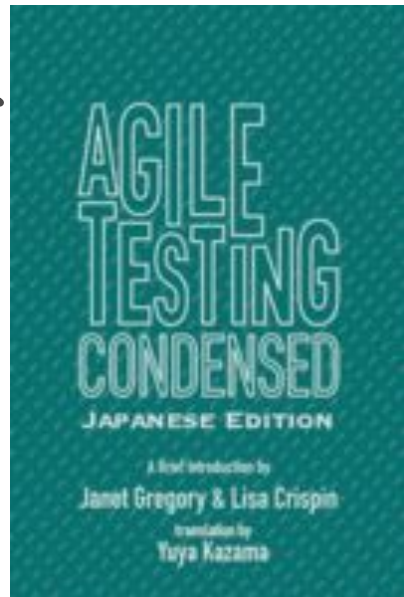


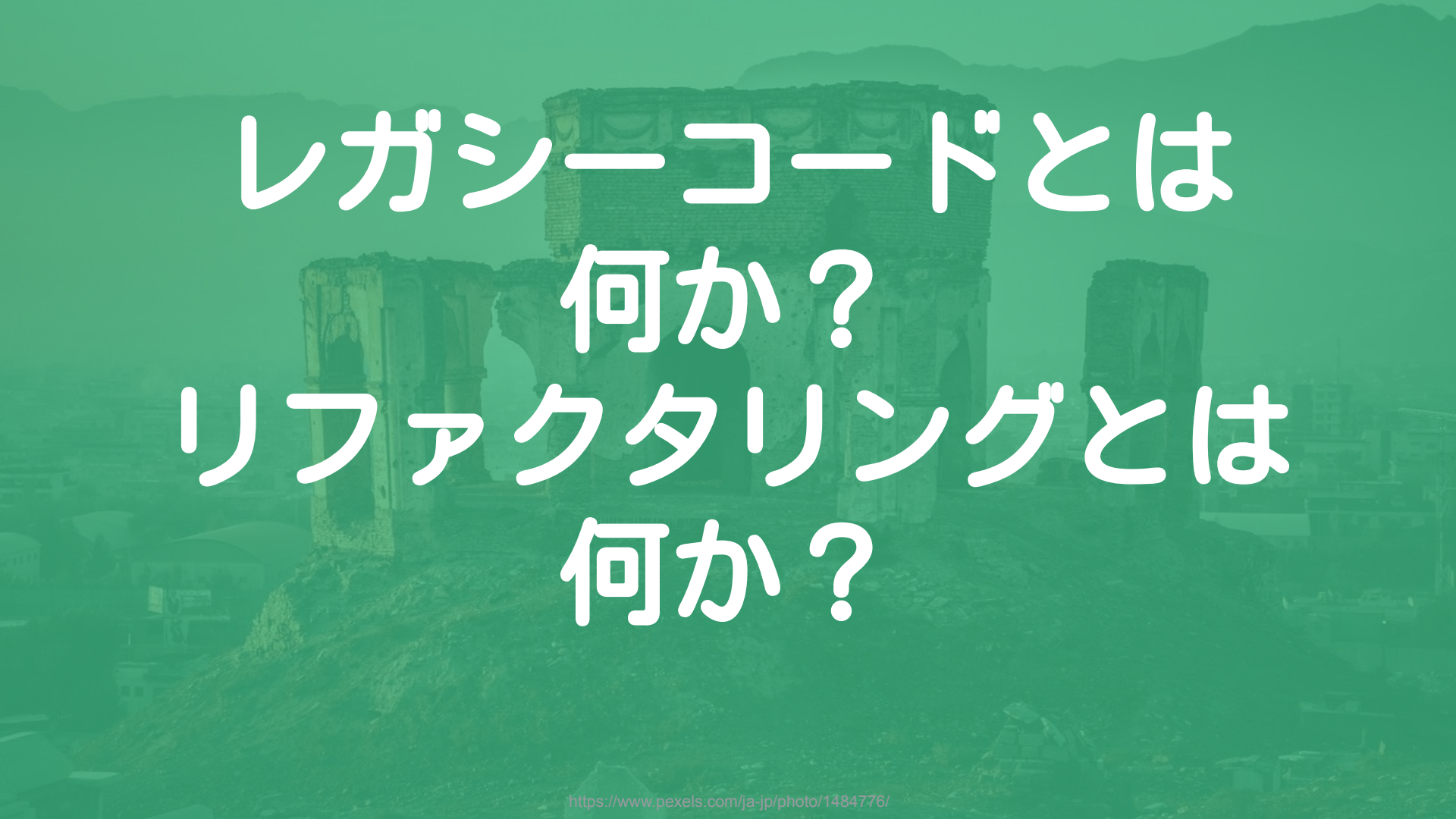
自己紹介

- 風間裕也（ブロッコリー）
- @nihonbuson
- 所属
 - 株式会社ビズリーチ（Visionalグループ）
 - QA基盤推進室 QA Evangelist
- 執筆活動
 - 『Agile Testing Condensed』 翻訳
 - 『テストコードの注入から始めるレガシーコードのリファクタリング』執筆中（技術同人誌）

SNS上の
アイコン

年末に
update !

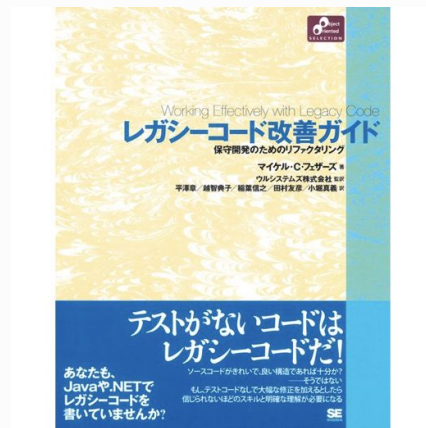




レガシーコードとは
何か？
リファクタリングとは
何か？

レガシーコードとは何か？

私にとって、レガシーコードとは、
単にテストのないコードです。



『レガシーコード改善ガイド』
「はじめに」より

システム変更を行う2つの方法

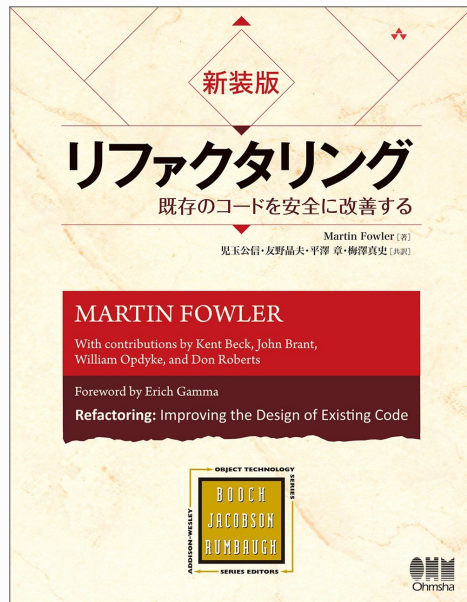
- リファクタリング…保護して変更する
→転落防止の網を張って変更する
- リライト…編集して祈る
→実行しようとする変更作業を
注意深く計画し、理解した上で変更する

『レガシーコード改善ガイド』
「第2章」より

リファクタリングの第一歩

リファクタリングを開始するとき、
最初にすることは常に同じです。
対象となるコードについてきちんと
した**テスト群を作りあげること**です。
(中略)

テストは大切で、きちっとした一連の
テストを用意すべきなのです。



『新装版リファクタリング』
「第1章」より

リファクタリングの第一歩

リファクタリングを開始する、
最初にすることは常に同じ。
対象となるコードは、
書いたテスト型
(中略)

書籍内にテストコードの
記載は無し

連の

9。



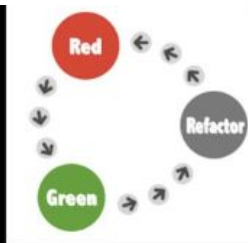
『新装版リファクタリング』
「第1章」より

レガシーコードの リファクタリング テクニック

TDDのサイクルを応用する

TDDのサイクル

1. 次の目標を考える
2. その目標を示すテストを書く
3. そのテストを実行して失敗させる(**Red**)
4. 目的のコードを書く
5. 2で書いたテストを成功させる(**Green**)
6. テストが通るままでリファクタリングを行う(**Refactor**)
7. 1～6を繰り返す



レガシーコードのリファクタリングの手順

1. **まずはテストを1つ**書いて実行する
2. テストを通す
3. テストが通ったままリファクタリングを行う
4. 次の目標を考える
5. 次の目標を達成するためのテストコードを書く
6. リファクタリングしたいと思った**部分のカバレッジを
満たすテストコード**が書けたことを確認する
7. カバレッジを満たしている部分から**逸脱しない
&テストが通ったまま**、リファクタリングを行う
8. 4～7を繰り返す

その他のリファクタリングテクニック

- 時には、エディタの機能を最大限に活用する

- 変数名のリネーム
- メソッドの切り出し
- メソッドの
クラス間移動
- if文の逆転

```
if(x==false) {  
    methodA();  
} else {  
    methodB();  
}
```



```
if(x==true) {  
    methodB();  
} else {  
    methodA();  
}
```

- リファクタリングの過程で新規クラスを作るときは、TDDのやり方を活用する

今回のお題

在庫更新システム「Gilded Rose」

- アイテムには下記の情報がある
 - Item名(name)
 - 販売期限(Sell in)
 - アイテムの価値(Quality)
- 1日の終わりには（更新すると）、
Sell inとQualityが1だけ小さくなる
- Sell inやQualityの値が上記の増減ルールとは異なる商品「Aged Brie」「Backstage passes」「Sulfuras」が存在する。
- 詳しい仕様書と対象コードは予稿集参照

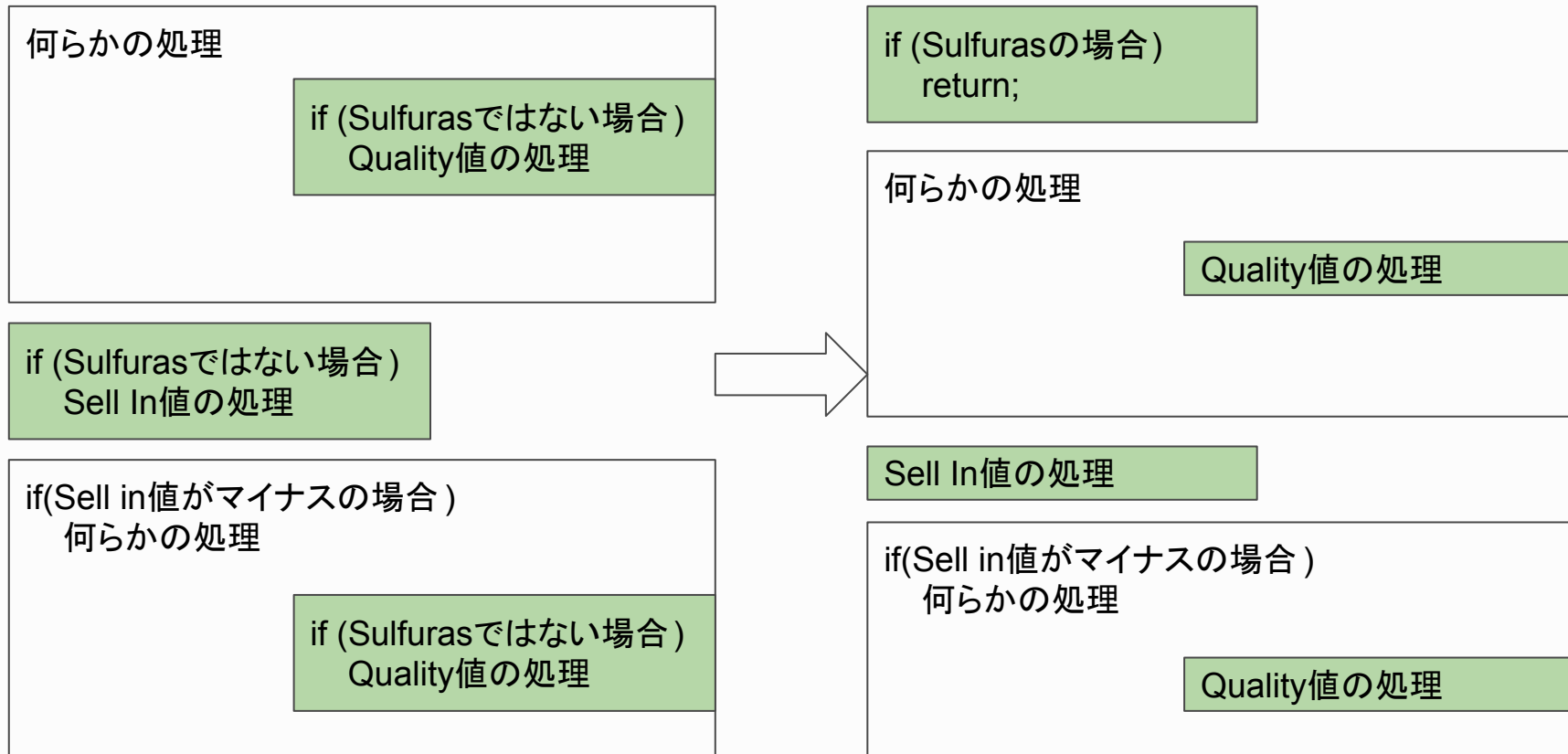
ライブコーディング中の諸注意

- ライブコーディングは事前録画です。
- プログラミングを知っている前提で説明します。
- Javaを多少知っている（何かが書いてあるかは分かる）前提で説明します。
- 気になったことがあれば、下記2つのハッシュタグを両方付けてツイートしてください。
 - #JaSST
 - #Legacy
- 当日用意されているSlidoでも質問を受け付けます。
 - 事前録画終了後の時間もしくは後日回答します。



講演中の補足資料

次の方針(Sulfuras処理の早期Return)



大きな方針(商品別の処理分岐)

if (Sulfurasの場合)
return;

何らかの処理

Sell In値の処理

if(Sell in値がマイナスの場合)
何らかの処理

if (Sulfurasの場合)
何らかの処理

if(AgedBrieの場合)
何らかの処理

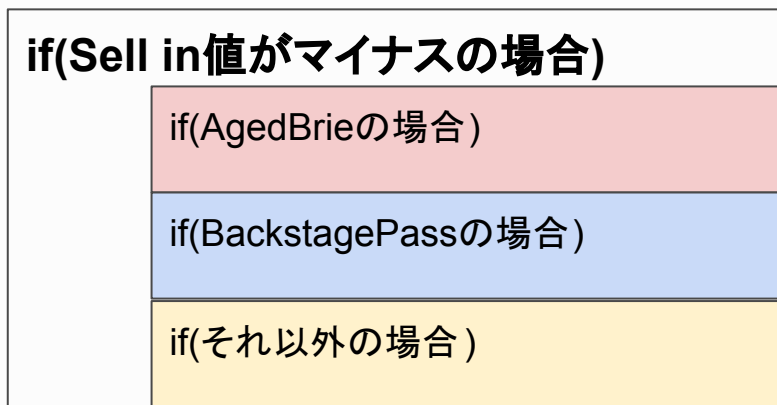
if(BackstagePassの場合)
何らかの処理

if(それ以外の場合)
何らかの処理

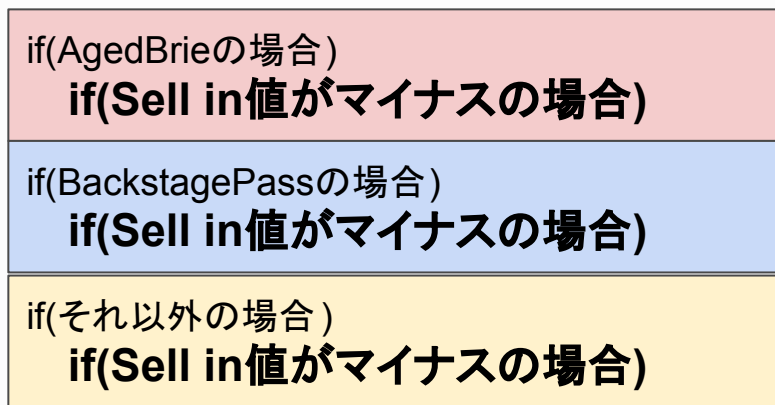
次の方針 (if構造の入れ替え)



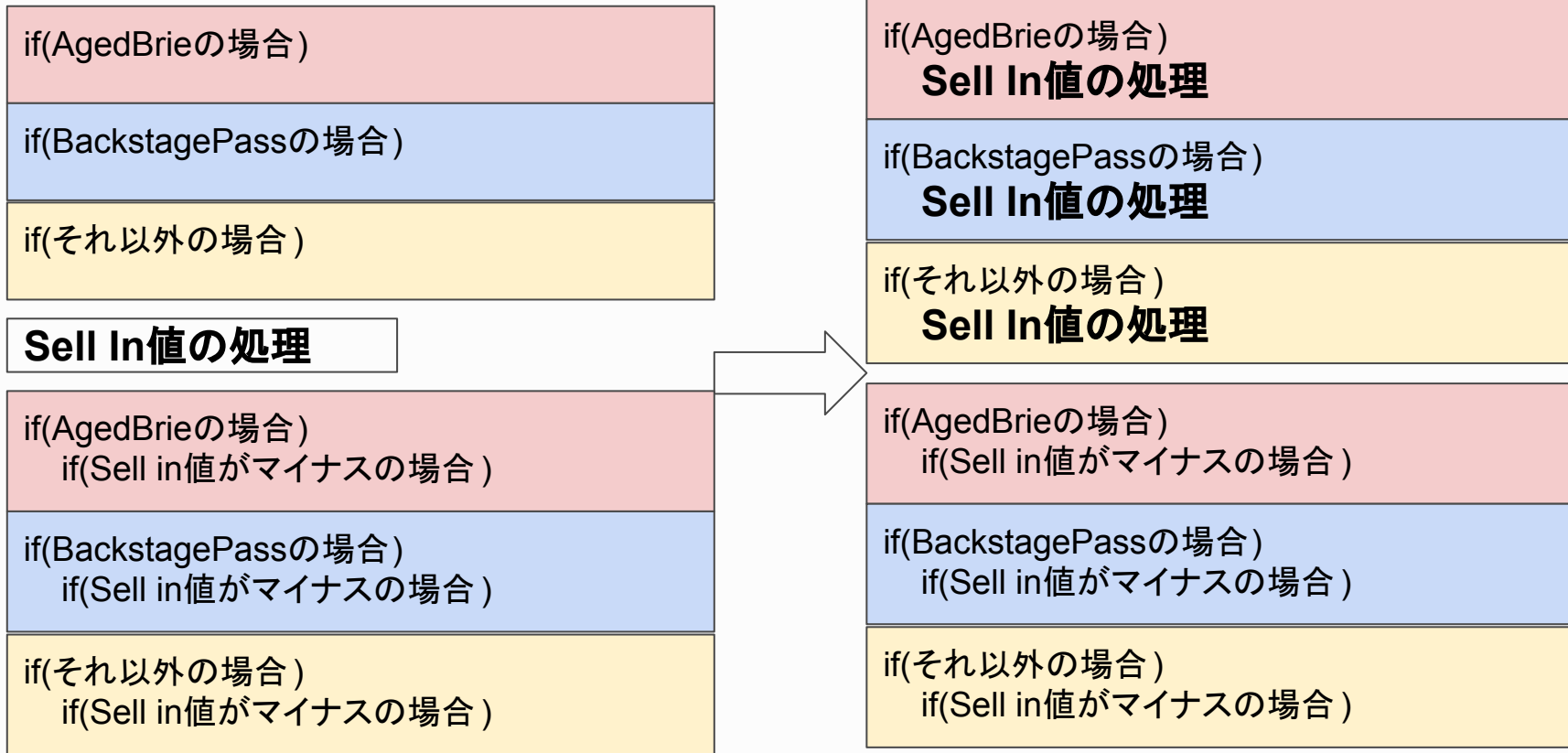
Sell In値の処理



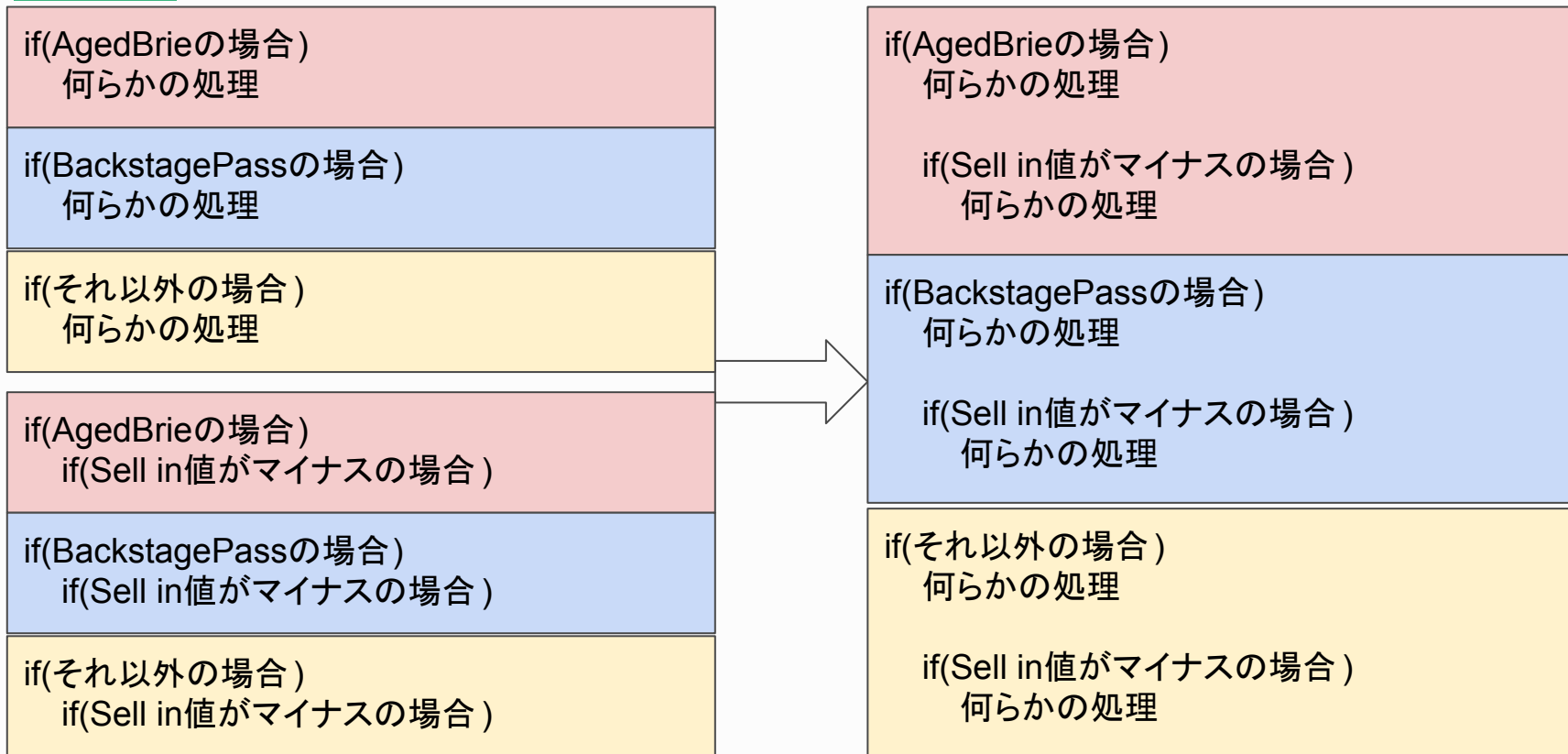
Sell In値の処理



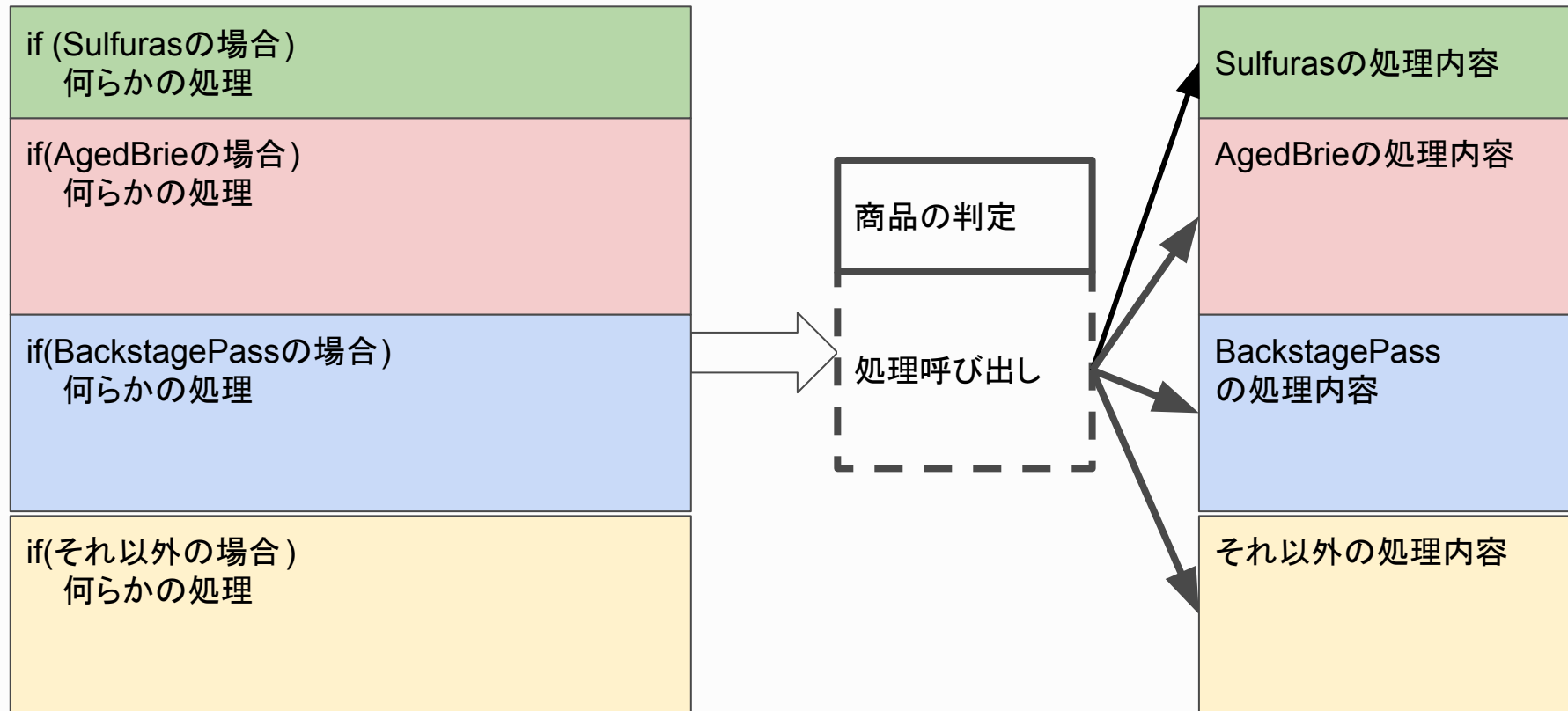
次の方針 (Sell in値の処理の移動)



次の方針(商品別の処理にまとめる)



大きな方針（インタフェースの検討）



当日いただいた質問

The background is a solid green color. Overlaid on this are several light green, semi-transparent question marks of various sizes and orientations. In the lower-left corner, there is a light green notepad with horizontal lines and a silver pen resting on it.

質問1

Q. 処理の分かりやすさに着目してリファクタリングをしていると思いますが、
処理の効率性（メモリ効率、CPU効率など）に着目してリファクタリングすることはありますか？

A. ケースバイケースです。
普段は着目することがあまりありませんが、
例えば大量のデータを扱うロジックの場所では、
着目すると思います。

質問2

Q. 次の目標設定して進める、ということはわかりました。
全体的なリファクタリングのゴールは
どこに設定するのがベストでしょうか。
(想像では、デザインパターンを使うところまで?)

A. リファクタリングは明確なゴールがないため、
大抵は時間で区切ったりします。
なお、今回はデザインパターンを頑張って使おうと
していません。リファクタリングした結果、
たまたまデザインパターンが適切だっただけです。

質問3

Q. フィクスチャやパラメタライズなどxUnitフレームワークの機能を積極的に使うことはお勧めでしょうか？

A. 最初はあまりオススメしません。

たしかに、パラメタライズを使うことで、短時間で大量のテストケースを作ることができます。

しかしその結果、テストの目的への意識が薄くなります。なぜこれらのテストケースを作ったのか分からなくなる可能性が高くなります。

最初は、なぜこのテストを作っているのか意識するために、基本に従ってテストケースを作することをオススメします。

質問4

Q. QAエンジニアとして今回の発表をチームに取り入れるにはやはりペアプロ（モブプロ）でしょうか。
（一緒にリファクタやテストコードを注入する）

A. リファクタリングをするには、
そのコードを書いた開発者が関わるのが
一番コストが低いので、
開発者と一緒にペアプロするのが良いと思います。

質問5

Q. 今回の発表のようなJavaでIntelliJの場合以外でも、例えばC#で、VSCodeでも同じ様に進められると認識しておりますが、あっていますでしょうか？

A. 合っています。
言語に依存する部分や、エディタの違いによってできる機能やできない機能の差はありますが、同様の方法でリファクタリングしていくことが可能です。