

Refactoring CSS

管理されたカオスへの道のり

Yuya Saito / Nikkei

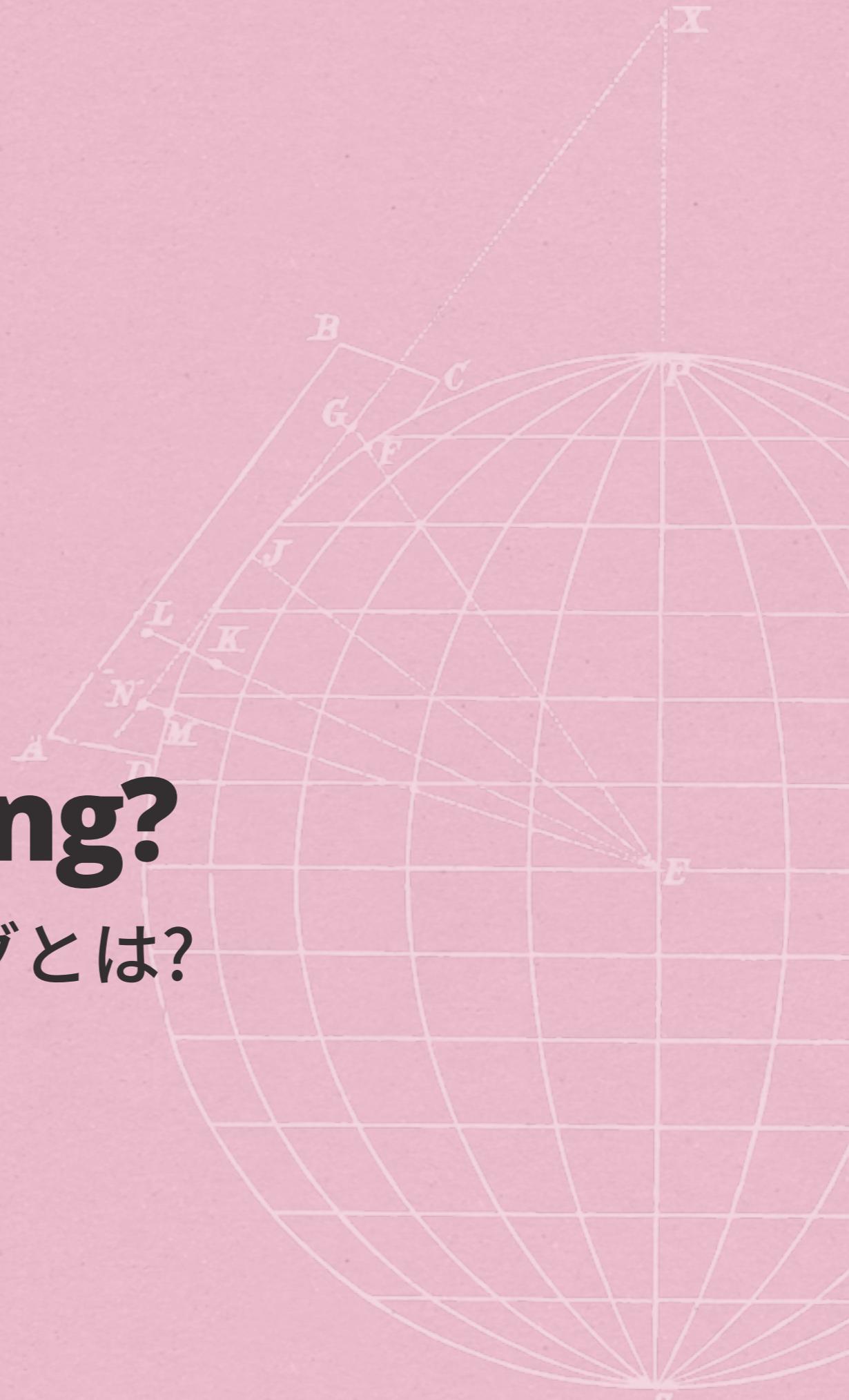
Hiroki Tani / tsukuruba

2017.02.25 @ Inside Frontend #1

Act I: Setup

What is, Why Refactoring?

リファクタリングとは?
なぜ必要なのか?



Ernest Hemingway

Novelist





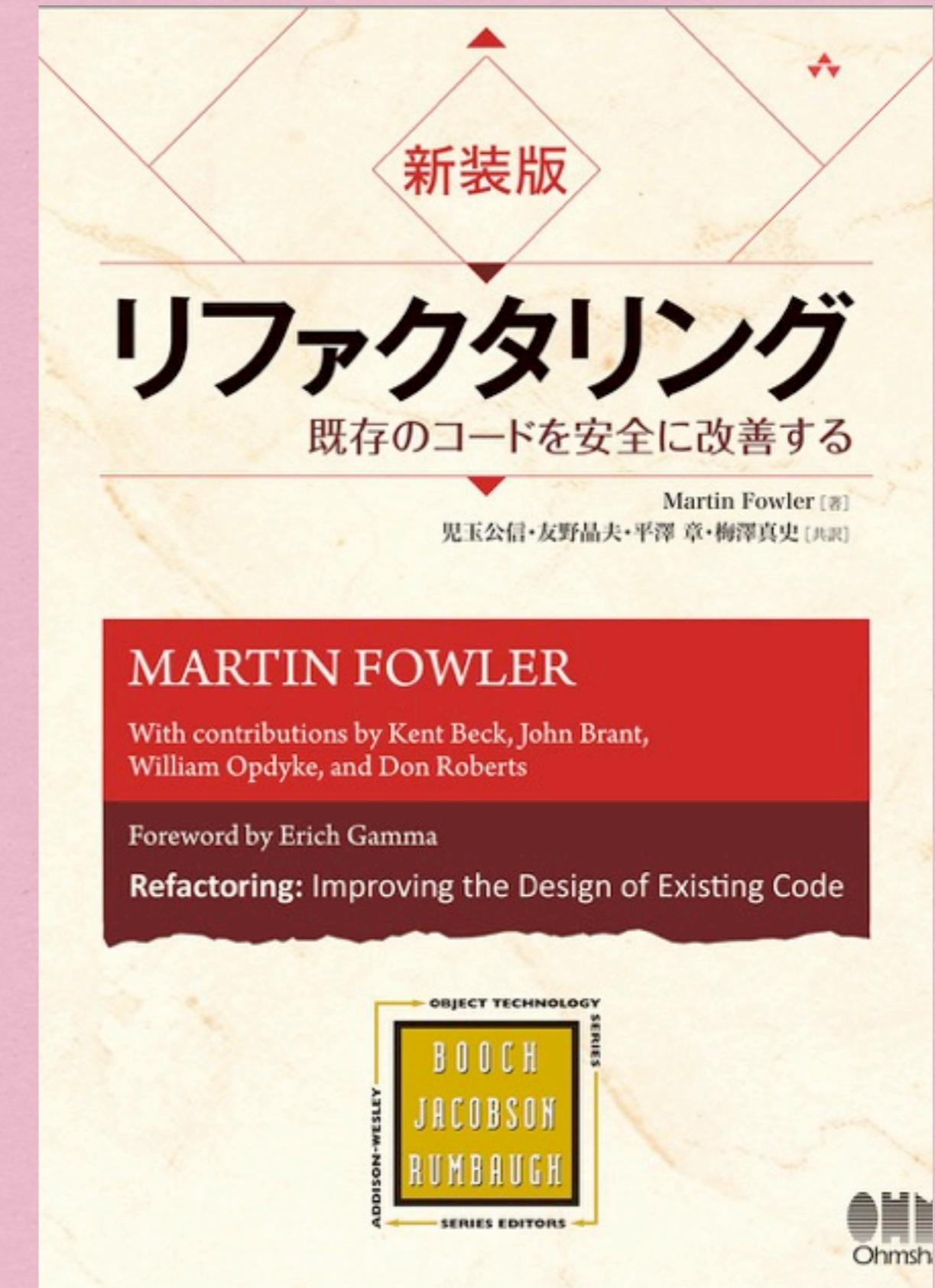
一番最初の下書きはどんな場合でも
イマイチなものだ。

— Ernest Hemingway

リファクタリング

by Martin Fowler

amzn.asia/6ohAyxd





リファクタリングとはソフトウェアの観測
できる外部的振る舞いを変えずに、
内部構造を改善するための統制された
テクニックである。

— Martin Fowler

**保守しやすく
拡張しやすい**

**CSSにおける
リファクタリングには
難しい場面が多くある**

**UIの変更頻度と変更速度
影響範囲の広さ
テストを書くことが難しい**

ROI...?



Act II: In Details

Refactoring When?

リファクタリングは何時するのか?



Always be Refactoring

**機能実装前、
バグ修正時**

コメントを残す

Code Smells

Rule of Three

1, 3, - - -
1, 3, 5, - - -



**Refactoring CSS doesn't mean rewriting
selectors w/ the n number of dashes and
underscores. Focus on what's between the curly
braces.**

— Mark Otto

Continuous Refactoring

技術的負債の 返済時

Refactoring How?

リファクタリングはどうやるのか?

**リファクタリング中は
それだけをする**

**問題は小さく碎いて
解決する**



By Brad Frost, Atomic Design

**元のソースコードは
消さない**

.RF_

outline: 1px dotted red;

all: initial;

```
.c-nav-primary {  
    font-size: 1rem;  
    font-family: sans-serif;  
}  
  
.c-nav-primary__item {  
    ...  
}  
  
/* Refactoring ... */  
.RF_c-nav-primary {  
    all: initial;  
  
    /* all: initial stops inheritance */  
    font-size: 2rem;  
}
```

The screenshot shows a browser window with the URL `jsbin.com/sakatab/edit?html,css,output`. The interface is divided into several sections:

- HTML:** Contains the following code:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width">
6   <title>Demo: all:initial and .RF</title>
7 </head>
8 <body>
9
10 <nav class="c-nav-primary RF_c-nav-primary">
11   <a href="#" class="c-nav-primary__item">I'm a
12     navigation item</a>
13   <a href="#" class="c-nav-primary__item">I'm a
14     navigation item</a>
15   <a href="#" class="c-nav-primary__item">I'm a
16     navigation item</a>
17 </body>
18 </html>
```
- CSS:** Contains the following code:

```
1 .c-nav-primary {
2   font-size: 1rem;
3   font-family: sans-serif;
4 }
5
6 .c-nav-primary__item {
7   color: black;
8   background: cornsilk;
9   display: inline-block;
10  padding: 0.5rem;
11  text-decoration: none;
12  width: 100%;
13 }
14
15 .c-nav-primary__item + .c-nav-primary__item {
16   margin-top: 0.5rem;
17 }
18
19 /* Refactoring... */
20 .RF_c-nav-primary {
21   all: initial;
22
23   /* all: initial effectively stops inheritance
24   */
25   font-size: 2rem;
26 }
```
- Output:** Displays four identical items, each containing the text "I'm a navigation item".
- Bottom right corner:** A small box labeled "Bin info".

`studiomohawk.jsbin.com/sakatab/edit?html,css,output`

ドキュメント

ドキュメンテーション ≠ スタイルガイド

// pattern2

// XL, L ⇒ M

⇒ S

// +-----+ +-TITLE-----+ +-----+ +-----+

// | IMG | +-TITLE-----+ | IMG | | IMG |

// | (OPT) | +-META-----+ | | | (OPT) |

// +-----+ +-EXCERPT----+ +-----+ +-----+

// +-TITLE-----+ +-0-4 LINE---+ +-TITLE-----+

// +-TITLE-----+ +-TITLE-----+

// +-META-----+ +-META-----+

// +-EXCERPT----+ +-EXCERPT----+

// +-0-4 LINE---+ +-0-4 LINE---+

```
// nui-card-scale--pattern3
// | breakpoint | font-size(px) | font-size(rem) | line-height |
// |-----|-----|-----|-----|
// | default | 14 | 0.875 | 1.50 |
// | M | 18 | 1.125 | 1.33 |
// | XL | 21 | 1.313 | 1.33 |
```

Refactoring Tips

リファクタリングのコツ

**技術的負債の
ご利用は計画的に**

技術的負債
≠
古いコード

技術的負債
≠
汚いコード

**戦略的で、意識的な
『ショートカット』**

**短期的な目標だけではなく
中長期的な視点を持つ**

負債を明確にする 設計

shame.css

by Harry Roberts

csswizardry.com/2013/04/shame-css

The screenshot shows a web browser window with a red header bar. The title bar reads 'csswizardry.com'. The main content area features a red 'css' logo at the top left. Below it, the date '17 April, 2013' is displayed. The main heading 'shame.css' is in red. A note below the heading says 'Update: I did a short interview about shame.css with .net magazine.' The main text discusses the idea of a stylesheet for 'nasty, hacky, quick-fix CSS' and includes a section titled 'The problem' with a note about using 'overflow: hidden;'. The bottom right corner contains a footer note about leaving hacks in code.

17 April, 2013

shame.css

Update: I did a short interview about shame.css with .net magazine.

Something [Chris Coyier](#), [Dave Rupert](#) and I [joked about recently](#) was the idea of a stylesheet dedicated to housing your nasty, hacky, quick-fix CSS. I've been thinking a lot more about it lately and I reckon it's actually a pretty good idea; here's why...

The problem

We all know that, no matter how hard we may try, sometimes we *do* need to use fixes, hacks and questionable techniques in our code. It happens, whether we like it or not.

From using a quick `overflow: hidden;` instead of working out what's actually broken our layout, to the odd `!important` to override some poor CSS, there are times where we need to use less than ideal CSS in order to meet deadlines, to get something working, or to fix pressing—or even live—bugs.

Whilst this isn't ideal, we *have* to do it from time to time; **all of us**.

The real problem, though, is that we rarely go back and tidy these things up. They fall through the cracks, get ignored, go unnoticed, and stay for good. This is something we do not do.

The problem with leaving hacks and nasty code is obvious; it's hacky and nasty.

コメントを確實に、 そして詳しく記載する

- コードベースのどこに関連しているのか?
- なぜ必要だったのか?
- どうやって問題を解決しているのか?
- 負債返済の条件

命名規則:
._ (アンダースコア)

(ちゃんと)
コメントを残す

Act III: Conclusion



残った疑問

- 気がつかない間に負債となってしまったものたちをどうするのか?
- リファクタリングの最大の武器、テストについて
- リファクタリングと政治

Ask US Anything

結びに

リファクタリングとは、
ひらめきを冷徹な頭で見直すこと。

The Long Goodbye

by Raymond Chandler

amzn.asia/hCsYlF4

ロング・グッドバイ
レイモンド・チャンドラー
村上春樹 訳

Raymond Chandler

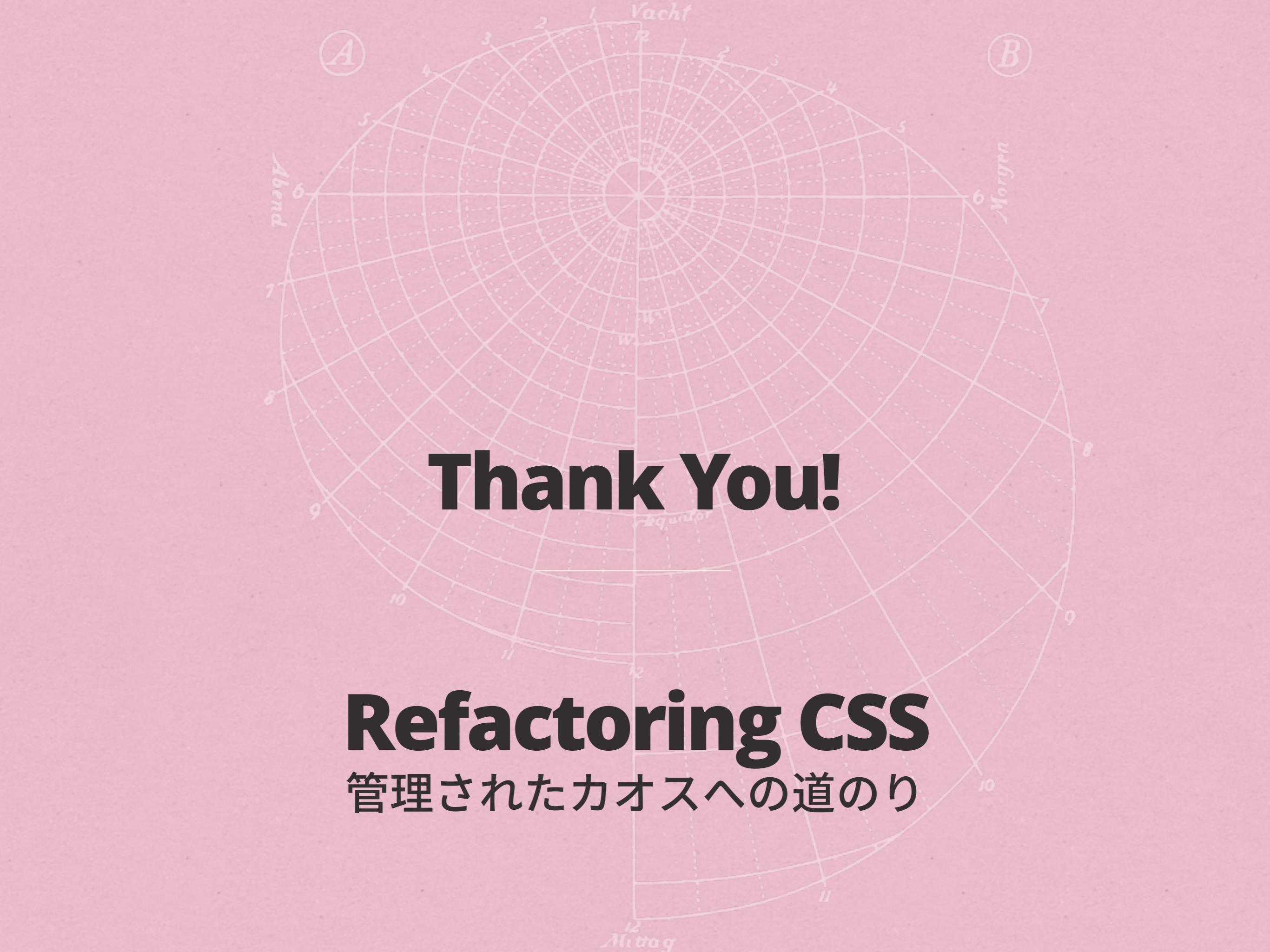
The Long Goodbye

早川書房



自分で自分に仕掛ける罠ほど
たちの悪い罠はない。

— Raymond Chandler



Thank You!

Refactoring CSS

管理されたカオスへの道のり