异步图书
www.epubit.com

Pearson

# 重构

## 改善既有代码的设计

（第2版）

[美] 马丁·福勒（Martin Fowler）著
熊节 林从羽 译

REFACTORING

Improving the Design of Existing Code
SECOND EDITION

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

1.1
1.2
1 1.3
1.1 1.3.1
1.2 1.3.2
1.3 1.3.3
1.4 statement 1.3.4
play 1.3.4.1
format 1.3.4.2
1.3.4.3
1.5 1.3.5
1.6 1.3.6
1.7 1.3.7
1.8 1.3.8
1.3.8.1
1.3.8.2
1.9 1.3.9
1.10 1.3.10
2 1.4
2.1 1.4.1
2.2 1.4.2
2.3 1.4.3
1.4.3.1
1.4.3.2
bug 1.4.3.3
1.4.3.4
2.4 1.4.4
1.4.4.1
1.4.4.2
1.4.4.3
1.4.4.4
1.4.4.5
1.4.4.6
1.4.4.7
1.4.4.8
2.5 1.4.5

1.4.5.1
1.4.5.2
1.4.5.3
1.4.5.4
1.4.5.5
1.4.5.6
2.6 YAGNI 1.4.6
2.7 1.4.7
2.8 1.4.8
2.9 1.4.9
2.10 1.4.10
2.11 1.4.11
3 1.5
3.1 Mysterious Name 1.5.1
3.2 Duplicated Code 1.5.2
3.3 Long Function 1.5.3
3.4 Long Parameter List 1.5.4
3.5 Global Data 1.5.5
3.6 Mutable Data 1.5.6
3.7 Divergent Change 1.5.7
3.8 Shotgun Surgery 1.5.8
3.9 Feature Envy 1.5.9
3.10 Data Clumps 1.5.10
3.11 Primitive Obsession 1.5.11
3.12 switch Repeated Switches 1.5.12
3.13 Loops 1.5.13
3.14 Lazy Element 1.5.14
3.15 Speculative Generality 1.5.15
3.16 Temporary Field 1.5.16
3.17 Message Chains 1.5.17
3.18 Middle Man 1.5.18
3.19 Insider Trading 1.5.19
3.20 Large Class 1.5.20
3.21 Alternative Classes with Different Interfaces 1.5.21
3.22 Data Class 1.5.22
3.23 Refused Bequest 1.5.23
3.24 Comments 1.5.24
4 1.6

4.1 1.6.1
4.2 1.6.2
4.3 1.6.3
4.4 1.6.4
4.5 1.6.5
4.6 1.6.6
4.7 1.6.7
5 1.7
5.1 1.7.1
5.2 1.7.2
6 1.8
6.1 Extract Function 1.8.1
1.8.1.1
1.8.1.2
1.8.1.3
1.8.1.4
1.8.1.5
6.2 Inline Function 1.8.2
1.8.2.1
1.8.2.2
1.8.2.3
6.3 Extract Variable 1.8.3
1.8.3.1
1.8.3.2
1.8.3.3
1.8.3.4
6.4 Inline Variable 1.8.4
1.8.4.1
1.8.4.2
6.5 Change Function Declaration 1.8.5
1.8.5.1
1.8.5.2
1.8.5.3
1.8.5.4
1.8.5.5
1.8.5.6
1.8.5.7
1.8.5.8

6.6 Encapsulate Variable 1.8.6
1.8.6.1
1.8.6.2
1.8.6.3
1.8.6.4
6.7 Rename Variable 1.8.7
1.8.7.1
1.8.7.2
1.8.7.3
1.8.7.4
6.8 Introduce Parameter Object 1.8.8
1.8.8.1
1.8.8.2
1.8.8.3
6.9 Combine Functions into Class 1.8.9
1.8.9.1
1.8.9.2
1.8.9.3
6.10 Combine Functions into Transform 1.8.10
1.8.10.1
1.8.10.2
1.8.10.3
6.11 Split Phase 1.8.11
1.8.11.1
1.8.11.2
1.8.11.3
7 1.9
7.1 Encapsulate Record 1.9.1
1.9.1.1
1.9.1.2
1.9.1.3
1.9.1.4
7.2 Encapsulate Collection 1.9.2
1.9.2.1
1.9.2.2
1.9.2.3
7.3 Replace Primitive with Object 1.9.3
1.9.3.1

1.9.3.2
1.9.3.3
7.4 Replace Temp with Query 1.9.4
1.9.4.1
1.9.4.2
1.9.4.3
7.5 Extract Class 1.9.5
1.9.5.1
1.9.5.2
1.9.5.3
7.6 Inline Class 1.9.6
1.9.6.1
1.9.6.2
1.9.6.3
7.7 Hide Delegate 1.9.7
1.9.7.1
1.9.7.2
1.9.7.3
7.8 Remove Middle Man 1.9.8
1.9.8.1
1.9.8.2
1.9.8.3
7.9 Substitute Algorithm 1.9.9
1.9.9.1
1.9.9.2
8 1.10
8.1 Move Function 1.10.1
1.10.1.1
1.10.1.2
1.10.1.3
1.10.1.4
8.2 Move Field 1.10.2
1.10.2.1
1.10.2.2
1.10.2.3
1.10.2.4
8.3 Move Statements into Function 1.10.3
1.10.3.1

1.10.3.2
1.10.3.3
8.4 Move Statements to Callers 1.10.4
1.10.4.1
1.10.4.2
1.10.4.3
8.5 Replace Inline Code with Function Call 1.10.5
1.10.5.1
1.10.5.2
8.6 Slide Statements 1.10.6
1.10.6.1
1.10.6.2
1.10.6.3
1.10.6.4
8.7 Split Loop 1.10.7
1.10.7.1
1.10.7.2
1.10.7.3
8.8 Replace Loop with Pipeline 1.10.8
1.10.8.1
1.10.8.2
1.10.8.3
8.9 Remove Dead Code 1.10.9
1.10.9.1
1.10.9.2
9 1.11
9.1 Split Variable 1.11.1
1.11.1.1
1.11.1.2
1.11.1.3
1.11.1.4
9.2 Rename Field 1.11.2
1.11.2.1
1.11.2.2
1.11.2.3
9.3 Replace Derived Variable with Query 1.11.3
1.11.3.1
1.11.3.2

| | | |
|---|---|---|
| | | 1.11.3.3 |
| | | 1.11.3.4 |
| 9.4 | Change Reference to Value | 1.11.4 |
| | | 1.11.4.1 |
| | | 1.11.4.2 |
| | | 1.11.4.3 |
| 9.5 | Change Value to Reference | 1.11.5 |
| | | 1.11.5.1 |
| | | 1.11.5.2 |
| | | 1.11.5.3 |
| 10 | | 1.12 |
| 10.1 | Decompose Conditional | 1.12.1 |
| | | 1.12.1.1 |
| | | 1.12.1.2 |
| | | 1.12.1.3 |
| 10.2 | Consolidate Conditional Expression | 1.12.2 |
| | | 1.12.2.1 |
| | | 1.12.2.2 |
| | | 1.12.2.3 |
| | | 1.12.2.4 |
| 10.3 | Replace Nested Conditional with Guard Clauses | 1.12.3 |
| | | 1.12.3.1 |
| | | 1.12.3.2 |
| | | 1.12.3.3 |
| | | 1.12.3.4 |
| 10.4 | Replace Conditional with Polymorphism | 1.12.4 |
| | | 1.12.4.1 |
| | | 1.12.4.2 |
| | | 1.12.4.3 |
| | | 1.12.4.4 |
| 10.5 | Introduce Special Case | 1.12.5 |
| | | 1.12.5.1 |
| | | 1.12.5.2 |
| | | 1.12.5.3 |
| | | 1.12.5.4 |
| | | 1.12.5.5 |
| 10.6 | Introduce Assertion | 1.12.6 |
| | | 1.12.6.1 |

1.12.6.2
1.12.6.3
11 API 1.13
11.1 Separate Query from Modifier 1.13.1
1.13.1.1
1.13.1.2
1.13.1.3
11.2 Parameterize Function 1.13.2
1.13.2.1
1.13.2.2
1.13.2.3
11.3 Remove Flag Argument 1.13.3
1.13.3.1
1.13.3.2
1.13.3.3
11.4 Preserve Whole Object 1.13.4
1.13.4.1
1.13.4.2
1.13.4.3
1.13.4.4
11.5 Replace Parameter with Query 1.13.5
1.13.5.1
1.13.5.2
1.13.5.3
11.6 Replace Query with Parameter 1.13.6
1.13.6.1
1.13.6.2
1.13.6.3
11.7 Remove Setting Method 1.13.7
1.13.7.1
1.13.7.2
1.13.7.3
11.8 Replace Constructor with Factory Function 1.13.8
1.13.8.1
1.13.8.2
1.13.8.3
11.9 Replace Function with Command 1.13.9
1.13.9.1

1.13.9.2
1.13.9.3
11.10 Replace Command with Function 1.13.10
1.13.10.1
1.13.10.2
1.13.10.3
12 1.14
12.1 Pull Up Method 1.14.1
1.14.1.1
1.14.1.2
1.14.1.3
12.2 Pull Up Field 1.14.2
1.14.2.1
1.14.2.2
12.3 Pull Up Constructor Body 1.14.3
1.14.3.1
1.14.3.2
1.14.3.3
12.4 Push Down Method 1.14.4
1.14.4.1
1.14.4.2
12.5 Push Down Field 1.14.5
1.14.5.1
1.14.5.2
12.6 Replace Type Code with Subclasses 1.14.6
1.14.6.1
1.14.6.2
1.14.6.3
1.14.6.4
12.7 Remove Subclass 1.14.7
1.14.7.1
1.14.7.2
1.14.7.3
12.8 Extract Superclass 1.14.8
1.14.8.1
1.14.8.2
1.14.8.3
12.9 Collapse Hierarchy 1.14.9

1.14.9.1
1.14.9.2
12.10 Replace Subclass with Delegate 1.14.10
1.14.10.1
1.14.10.2
1.14.10.3
1.14.10.4
12.11 Replace Superclass with Delegate 1.14.11
1.14.11.1
1.14.11.2
1.14.11.3

# book-refactoring2


- : https://book-refactoring2.ifmicro.com
- : pdf, epub, mobi

/

1.
2.

```
$ git clone https://github.com/MwumLi/book-refactoring2.git
$ npm i
```

, , `npm install`

:

```
$ npm run build
```

`_book/` ,

, `mobi` , `epub` `pdf` :

```
$ npm run ebook
```

: calibre, gitbook

- Node.js ^10.x - ^11.x LTS
- gitbook ^3.x:

, pr

NxeedGoto/Refactoring2-zh, , gitbook

: -

: Martin Fowler

override

“ ” “ ”

“ ”

6

Kent Beck

1 Kent “ ”

1 18

refactoring

“ ”

“ ”

“ ”

“ - ”

1

2 3 Kent Beck “ ” “ ”

4

5 —— 20 90

—— 154 ——

# JavaScript

JavaScript

JavaScript JavaScript

JavaScript JavaScript 1 Java Java

“ ”

JavaScript JavaScript

“ ”“ ”“ ” JavaScript

JavaScript “ JavaScript ”

JavaScript JavaScript promise async/await

JavaScript

——

JavaScript

- 1
-
- 3 “ ”
-

106 154

20 90 1

Ward Cunningham Kent Beck
Kent

Ralph Johnson UIUC - Ralph
Bill Opdyke John Brant Don Roberts
Refactoring Browser Smalltalk

1
IDE

1 Kent Beck
“ ” 1 ——

Arlo
Belshee Avdi Grimm Beth Anders-Beck Bill Wake Brian Guthrie Brian Marick Chad Wathington Dave Farley David Rice Don Roberts Fred George Giles Alexander Greg Doench Hugo Corbucci Ivan Moore James Shore Jay Fields Jessica Kerr Joshua Kerievsky Kevlin Henney Luciano Ramalho Marcos Brizeno Michael Feathers Patrick Kua Pete Hodgson Rebecca Parsons Trisha Gee

Beth Anders-Beck James Shore Pete Hodgson JavaScript

William Chargin Michael Hunger
Bob Martin Scott Davis Bill Wake

ThoughtWorks ThoughtWorks
Rebecca Parsons CTO

Greg Doench Julie Nahil Dmitry Kirsanov
Alina Kirsanova

# 1

100

“ ”

## 1.1

1 “ ”

customer audience

tragedy comedy “ ” volume credit

——

JSON

### plays.json...

```
{
  "hamlet": { "name": "Hamlet", "type": "tragedy" },
  "as-like": { "name": "As You Like It", "type": "comedy" },
  "othello": { "name": "Othello", "type": "tragedy" }
}
```

JSON

### invoices.json...

```
[
  {
    "customer": "BigCo",
    "performances": [
      {
        "playID": "hamlet",
        "audience": 55
      },
      {
        "playID": "as-like",
        "audience": 35
      },
      {
        "playID": "othello",
        "audience": 40
      }
    ]
  }
]
```

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    const play = plays[perf.playID];
    let thisAmount = 0;

    switch (play.type) {
    case "tragedy":
      thisAmount = 40000;
      if (perf.audience > 30) {
        thisAmount += 1000 * (perf.audience - 30);
      }
      break;
    case "comedy":
      thisAmount = 30000;
      if (perf.audience > 20) {
        thisAmount += 10000 + 500 * (perf.audience - 20);
      }
      thisAmount += 300 * perf.audience;
      break;
    default:
        throw new Error(`unknown type: ${play.type}`);
    }

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === play.type) volumeCredits += Math.floor(perf.audience / 5);

    // print line for this order
    result += ` ${play.name}: ${format(thisAmount/100)} (${perf.audience} seats
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
}
```

invoices.json  plays.json

```
Statement for BigCo
  Hamlet: $650.00 (55 seats)
  As You Like It: $580.00 (35 seats)
  Othello: $500.00 (40 seats)
Amount owed is $1,730.00
You earned 47 credits
```

# 1.2

____ ____

“ ”

bug

> **Tip**

HTML result

HTML

6

statement statement HTML

# 1.3

bug

____

statement invoice statement

statement

> **Tip**

bug —— ——

4

## 1.4 statement

switch

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    const play = plays[perf.playID];
    let thisAmount = 0;

    switch (play.type) {
    case "tragedy":
      thisAmount = 40000;
      if (perf.audience > 30) {
        thisAmount += 1000 * (perf.audience - 30);
      }
      break;
    case "comedy":
      thisAmount = 30000;
      if (perf.audience > 20) {
        thisAmount += 10000 + 500 * (perf.audience - 20);
      }
      thisAmount += 300 * perf.audience;
      break;
    default:
        throw new Error(`unknown type: ${play.type}`);
    }

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === play.type) volumeCredits += Math.floor(perf.audience / 5);

    // print line for this order
    result += ` ${play.name}: ${format(thisAmount/100)} (${perf.audience} seats
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
}
```

Ward Cunningham

amountFor(performance)

perf play thisAmount 3
——thisAmount

## function statement...

```
function amountFor(perf, play) {
  let thisAmount = 0;
  switch (play.type) {
  case "tragedy":
    thisAmount = 40000;
    if (perf.audience > 30) {
      thisAmount += 1000 * (perf.audience - 30);
    }
    break;
  case "comedy":
    thisAmount = 30000;
    if (perf.audience > 20) {
      thisAmount += 10000 + 500 * (perf.audience - 20);
    }
    thisAmount += 300 * perf.audience;
    break;
  default:
      throw new Error(`unknown type: ${play.type}`);
  }
  return thisAmount;
}
```

“ function xxx ”

statement thisAmount

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                       { style: "currency", currency: "USD",
                         minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    const play = plays[perf.playID];
    let thisAmount = amountFor(perf, play);

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === play.type) volumeCredits += Math.floor(perf.audience / 5);

    // print line for this order
    result += ` ${play.name}: ${format(thisAmount/100)} (${perf.audience} seats
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

---

> **Tip**
>
> “ ” JavaScript JavaScript Babel

JavaScript amountFor statement

> **Tip**
>
> git mercurial push commit

106 Java IDE JavaScript

106 thisAmount result

## function statement...

```
function amountFor(perf, play) {
  let result = 0;
  switch (play.type) {
  case "tragedy":
    result = 40000;
    if (perf.audience > 30) {
      result += 1000 * (perf.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (perf.audience > 20) {
      result += 10000 + 500 * (perf.audience - 20);
    }
    result += 300 * perf.audience;
    break;
  default:
      throw new Error(`unknown type: ${play.type}`);
  }
  return result;
}
```

“result” ——

## function statement...

```
function amountFor(aPerformance, play) {
  let result = 0;
  switch (play.type) {
  case "tragedy":
    result = 40000;
    if (aPerformance.audience > 30) {
      result += 1000 * (aPerformance.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (aPerformance.audience > 20) {
      result += 10000 + 500 * (aPerformance.audience - 20);
    }
    result += 300 * aPerformance.audience;
    break;
  default:
      throw new Error(`unknown type: ${play.type}`);
  }
  return result;
}
```

JavaScript
Kent Beck [Beck SBPP]

> **Tip**

play

## play

amountFor aPerformance play performance
amountFor play
178

## function statement...

```
function playFor(aPerformance) {
  return plays[aPerformance.playID];
}
```

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    const play = playFor(perf);
    let thisAmount = amountFor(perf, play);

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === play.type) volumeCredits += Math.floor(perf.audience / 5);

    // print line for this order
    result += ` ${play.name}: ${format(thisAmount/100)} (${perf.audience} seats
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

123 play

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    const play = playFor(perf);
    let thisAmount = amountFor(perf, playFor(perf));

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === playFor(perf).type) volumeCredits += Math.floor(perf.audie

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(thisAmount/100)} (${perf.audier
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

amountFor 124 play amountFor

## function statement...

```
function amountFor(aPerformance, play) {
  let result = 0;
  switch (playFor(aPerformance).type) {
  case "tragedy":
    result = 40000;
    if (aPerformance.audience > 30) {
      result += 1000 * (aPerformance.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (aPerformance.audience > 20) {
      result += 10000 + 500 * (aPerformance.audience - 20);
    }
    result += 300 * aPerformance.audience;
    break;
  default:
      throw new Error(`unknown type: ${playFor(aPerformance).type}`);
  }
  return result;
}
```

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    let thisAmount = amountFor(perf , playFor(perf) );

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === playFor(perf).type) volumeCredits += Math.floor(perf.audie

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(thisAmount/100)} (${perf.audier
    totalAmount += thisAmount;
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

## function statement...

```
function amountFor(aPerformance , play ) {
  let result = 0;
  switch (playFor(aPerformance).type) {
  case "tragedy":
    result = 40000;
    if (aPerformance.audience > 30) {
      result += 1000 * (aPerformance.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (aPerformance.audience > 20) {
      result += 10000 + 500 * (aPerformance.audience - 20);
    }
    result += 300 * aPerformance.audience;
    break;
  default:
      throw new Error(`unknown type: ${playFor(aPerformance).type}`);
  }
  return result;
}
```

play 1 3

amountFor 123

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                       { style: "currency", currency: "USD",
                         minimumFractionDigits: 2 }).format;
    for (let perf of invoice.performances) {

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === playFor(perf).type) volumeCredits += Math.floor(perf.audie

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(amountFor(perf)/100)} (${perf.a
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

statement

- ...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {

    // add volume credits
    volumeCredits += Math.max(perf.audience - 30, 0);
    // add extra credit for every ten comedy attendees
    if ("comedy" === playFor(perf).type) volumeCredits += Math.floor(perf.audie

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(amountFor(perf)/100)} (${perf.a
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

play

perf volumeCredits
volumeCredits

## function statement...

```
function volumeCreditsFor(perf) {
  let volumeCredits = 0;
  volumeCredits += Math.max(perf.audience - 30, 0);
  if ("comedy" === playFor(perf).type)
    volumeCredits += Math.floor(perf.audience / 5);
  return volumeCredits;
}
```

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(amountFor(perf)/100)} (${perf.a
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

## function statement...

```
function volumeCreditsFor(aPerformance) {
  let result = 0;
  result += Math.max(aPerformance.audience - 30, 0);
  if ("comedy" === playFor(aPerformance).type)
    result += Math.floor(aPerformance.audience / 5);
  return result;
}
```

### format

statement

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  const format = new Intl.NumberFormat("en-US",
                        { style: "currency", currency: "USD",
                          minimumFractionDigits: 2 }).format;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(amountFor(perf)/100)} (${perf.a
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

format

“ ”

## function statement...

```
function format(aNumber) {
  return new Intl.NumberFormat("en-US", {
    style: "currency",
    currency: "USD",
    minimumFractionDigits: 2,
  }).format(aNumber);
}
```

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);

    // print line for this order
    result += ` ${playFor(perf).name}: ${format(amountFor(perf)/100)} (${perf.a
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${format(totalAmount/100)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

> **Tip**

——format formatAsUSD

124

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);

    // print line for this order
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
    totalAmount += amountFor(perf);
  }
  result += `Amount owed is ${usd(totalAmount)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

## function statement...

```
function usd(aNumber) {
  return new Intl.NumberFormat("en-US", {
    style: "currency",
    currency: "USD",
    minimumFractionDigits: 2,
  }).format(aNumber / 100);
}
```

100

volumeCredits 227 volumeCredits

•••

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let volumeCredits = 0;
  let result = `Statement for ${invoice.customer}\n`;

  for (let perf of invoice.performances) {

    // print line for this order
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
    totalAmount += amountFor(perf);
  }
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);
  }

result += `Amount owed is ${usd(totalAmount)}\n`;
result += `You earned ${volumeCredits} credits\n`;
return result;
```

223

top level…

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {

    // print line for this order
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
    totalAmount += amountFor(perf);
  }
  let volumeCredits = 0;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);
  }
  result += `Amount owed is ${usd(totalAmount)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

volumeCredits 178 106

## function statement...

```
function totalVolumeCredits() {
  let volumeCredits = 0;
  for (let perf of invoice.performances) {
    volumeCredits += volumeCreditsFor(perf);
  }
  return volumeCredits;
}
```

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {

    // print line for this order
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
    totalAmount += amountFor(perf);
  }
  let volumeCredits = totalVolumeCredits();
  result += `Amount owed is ${usd(totalAmount)}\n`;
  result += `You earned ${volumeCredits} credits\n`;
  return result;
```

123 totalVolumeCredits

...

```
function statement (invoice, plays) {
  let totalAmount = 0;
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {

    // print line for this order
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
    totalAmount += amountFor(perf);
  }

  result += `Amount owed is ${usd(totalAmount)}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;
```

---

“ ” “ ”

---

volumeCredits 4

- 227
- 223
- 106
- 123

---

totalAmount
totalAmount

## function statement...

```
function appleSauce() {
  let totalAmount = 0;
  for (let perf of invoice.performances) {
    totalAmount += amountFor(perf);
  }
  return totalAmount;
}
```

...

```
function statement (invoice, plays) {
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  let totalAmount = appleSauce();

  result += `Amount owed is ${usd(totalAmount)}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;
```

totalAmount

...

```
function statement (invoice, plays) {
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;
```

## function statement...

```
function totalAmount() {
  let totalAmount = 0;
  for (let perf of invoice.performances) {
    totalAmount += amountFor(perf);
  }
  return totalAmount;
}
```

## function statement...

```
function totalAmount() {
  let result = 0;
  for (let perf of invoice.performances) {
    result += amountFor(perf);
  }
  return result;
}
function totalVolumeCredits() {
  let result = 0;
  for (let perf of invoice.performances) {
    result += volumeCreditsFor(perf);
  }
  return result;
}
```

# 1.5

```
function statement (invoice, plays) {
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;

  function totalAmount() {
    let result = 0;
    for (let perf of invoice.performances) {
      result += amountFor(perf);
    }
    return result;
  }
  function totalVolumeCredits() {
    let result = 0;
    for (let perf of invoice.performances) {
      result += volumeCreditsFor(perf);
    }
    return result;
  }
  function usd(aNumber) {
    return new Intl.NumberFormat("en-US",
                         { style: "currency", currency: "USD",
                           minimumFractionDigits: 2 }).format(aNumber/100);
  }
  function volumeCreditsFor(aPerformance) {
    let result = 0;
    result += Math.max(aPerformance.audience - 30, 0);
    if ("comedy" === playFor(aPerformance).type) result += Math.floor(aPerforma
    return result;
  }
  function playFor(aPerformance) {
    return plays[aPerformance.playID];
  }
  function amountFor(aPerformance) {
    let result = 0;
    switch (playFor(aPerformance).type) {
    case "tragedy":
      result = 40000;
      if (aPerformance.audience > 30) {
        result += 1000 * (aPerformance.audience - 30);
      }
      break;
    case "comedy":
      result = 30000;
      if (aPerformance.audience > 20) {
        result += 10000 + 500 * (aPerformance.audience - 20);
      }
```

```
      result += 300 * aPerformance.audience;
      break;
    default:
      throw new Error(`unknown type: ${playFor(aPerformance).type}`);
    }
    return result;
  }
}
```

statement 7

# 1.6

HTML 7 HTML
HTML

154 HTML

154 106 statement
renderPlainText

```
function statement (invoice, plays) {
  return renderPlainText(invoice, plays);
}

function renderPlainText(invoice, plays) {
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;

function totalAmount() {...}
  function totalVolumeCredits() {...}
  function usd(aNumber) {...}
  function volumeCreditsFor(aPerformance) {...}
  function playFor(aPerformance) {...}
  function amountFor(aPerformance) {...}
```

renderPlainText

```
function statement (invoice, plays) {
  const statementData = {};
  return renderPlainText(statementData, invoice, plays);
}

function renderPlainText(data, invoice, plays) {
  let result = `Statement for ${invoice.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;

function totalAmount() {...}
  function totalVolumeCredits() {...}
  function usd(aNumber) {...}
  function volumeCreditsFor(aPerformance) {...}
  function playFor(aPerformance) {...}
  function amountFor(aPerformance) {...}
```

renderPlainText statement
renderPlainText data

customer

```
function statement (invoice, plays) {
  const statementData = {};
  statementData.customer = invoice.customer;
  return renderPlainText(statementData, invoice, plays);
}

function renderPlainText(data, invoice, plays) {
  let result = `Statement for ${data.customer}\n`;
  for (let perf of invoice.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;
```

performances renderPlainText invoice

•••

```
function statement (invoice, plays) {
  const statementData = {};
  statementData.customer = invoice.customer;
  statementData.performances = invoice.performances;
  return renderPlainText(statementData, plays);
}

function renderPlainText(data, plays) {
  let result = `Statement for ${data.customer}\n`;
  for (let perf of data.performances) {
    result += ` ${playFor(perf).name}: ${usd(amountFor(perf))} (${perf.audience
  }
  result += `Amount owed is ${usd(totalAmount())}\n`;
  result += `You earned ${totalVolumeCredits()} credits\n`;
  return result;
```

## function renderPlainText...

```
function totalAmount() {
  let result = 0;
  for (let perf of data.performances) {
    result += amountFor(perf);
  }
  return result;
}
function totalVolumeCredits() {
  let result = 0;
  for (let perf of data.performances) {
    result += volumeCreditsFor(perf);
  }
  return result;
}
```

“ ” play aPerformance

```
function statement (invoice, plays) {
  const statementData = {};
  statementData.customer = invoice.customer;
  statementData.performances = invoice.performances.map(enrichPerformance);
  return renderPlainText(statementData, plays);

  function enrichPerformance(aPerformance) {
    const result = Object.assign({}, aPerformance);
    return result;
  }
}
```

aPerformance

immutable ——

> **Tip**
>
> JavaScript result = Object.assign({}, aPerformance)
>
> JavaScript ::

play playFor statement 198

## function statement...

```
function enrichPerformance(aPerformance) {
  const result = Object.assign({}, aPerformance);
  result.play = playFor(result);
  return result;
}

function playFor(aPerformance) {
  return plays[aPerformance.playID];
}
```

renderPlainText playFor

## function renderPlainText...

```
let result = `Statement for ${data.customer}\n`;
for (let perf of data.performances) {
  result += ` ${perf.play.name}: ${usd(amountFor(perf))} (${perf.audience} seat
}
result += `Amount owed is ${usd(totalAmount())}\n`;
result += `You earned ${totalVolumeCredits()} credits\n`;
return result;

function volumeCreditsFor(aPerformance) {
  let result = 0;
  result += Math.max(aPerformance.audience - 30, 0);
  if ("comedy" === aPerformance.play.type) result += Math.floor(aPerformance.au
  return result;
}

function amountFor(aPerformance){
  let result = 0;
  switch (aPerformance.play.type) {
  case "tragedy":
    result = 40000;
    if (aPerformance.audience > 30) {
      result += 1000 * (aPerformance.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (aPerformance.audience > 20) {
      result += 10000 + 500 * (aPerformance.audience - 20);
    }
    result += 300 * aPerformance.audience;
    break;
  default:
    throw new Error(`unknown type: ${aPerformance.play.type}`);
  }
  return result;
}
```

amountFor

## function statement...

```
function enrichPerformance(aPerformance) {
  const result = Object.assign({}, aPerformance);
  result.play = playFor(result);
  result.amount = amountFor(result);
  return result;
}

function amountFor(aPerformance) {...}
```

## function renderPlainText...

```
let result = `Statement for ${data.customer}\n`;
for (let perf of data.performances) {
  result += ` ${perf.play.name}: ${usd(perf.amount)} (${
    perf.audience
  } seats)\n`;
}
result += `Amount owed is ${usd(totalAmount())}\n`;
result += `You earned ${totalVolumeCredits()} credits\n`;
return result;

function totalAmount() {
  let result = 0;
  for (let perf of data.performances) {
    result += perf.amount;
  }
  return result;
}
```

## function statement...

```
function enrichPerformance(aPerformance) {
  const result = Object.assign({}, aPerformance);
  result.play = playFor(result);
  result.amount = amountFor(result);
  result.volumeCredits = volumeCreditsFor(result);
  return result;
}

function volumeCreditsFor(aPerformance) {...}
```

## function renderPlainText...

```
function totalVolumeCredits() {
  let result = 0;
  for (let perf of data.performances) {
    result += perf.volumeCredits;
  }
  return result;
}
```

statement

## function statement...

```
  const statementData = {};
statementData.customer = invoice.customer;
statementData.performances = invoice.performances.map(enrichPerformance);
statementData.totalAmount = totalAmount(statementData);
statementData.totalVolumeCredits = totalVolumeCredits(statementData);
return renderPlainText(statementData, plays);

 function totalAmount(data) {...}
   function totalVolumeCredits(data) {...}
```

## function renderPlainText...

```
let result = `Statement for ${data.customer}\n`;
for (let perf of data.performances) {
  result += ` ${perf.play.name}: ${usd(perf.amount)} (${
    perf.audience
  } seats)\n`;
}
result += `Amount owed is ${usd(data.totalAmount)}\n`;
result += `You earned ${data.totalVolumeCredits} credits\n`;
return result;
```

statementData

231

## function renderPlainText...

```
  function totalAmount(data) {
  return data.performances
    .reduce((total, p) => total + p.amount, 0);
}
function totalVolumeCredits(data) {
  return data.performances
    .reduce((total, p) => total + p.volumeCredits, 0);
}
```

...

```
  function statement (invoice, plays) {
  return renderPlainText(createStatementData(invoice, plays));
}

function createStatementData(invoice, plays) {
  const statementData = {};
  statementData.customer = invoice.customer;
  statementData.performances = invoice.performances.map(enrichPerformance);
  statementData.totalAmount = totalAmount(statementData);
  statementData.totalVolumeCredits = totalVolumeCredits(statementData);
  return statementData;
```

## statement.js...

```
import createStatementData from "./createStatementData.js";
```

## createStatementData.js...

```
export default function createStatementData(invoice, plays) {
const result = {};
result.customer = invoice.customer;
result.performances = invoice.performances.map(enrichPerformance);
result.totalAmount = totalAmount(result);
result.totalVolumeCredits = totalVolumeCredits(result);
return result;

function enrichPerformance(aPerformance) {...}
  function playFor(aPerformance) {...}
  function amountFor(aPerformance) {...}
  function volumeCreditsFor(aPerformance) {...}
  function totalAmount(data) {...}
  function totalVolumeCredits(data) {...}
```

HTML

### statement.js...

```
function htmlStatement (invoice, plays) {
  return renderHtml(createStatementData(invoice, plays));
}
function renderHtml (data) {
  let result = `<h1>Statement for ${data.customer}</h1>\n`;
  result += "<table>\n";
  result += "<tr><th>play</th><th>seats</th><th>cost</th></tr>";
  for (let perf of data.performances) {
    result += `  <tr><td>${perf.play.name}</td><td>${perf.audience}</td>`;
    result += `<td>${usd(perf.amount)}</td></tr>\n`;
  }
  result += "</table>\n";
  result += `<p>Amount owed is <em>${usd(data.totalAmount)}</em></p>\n`;
  result += `<p>You earned <em>${data.totalVolumeCredits}</em> credits</p>\n`;
  return result;
}

function usd(aNumber) {...}
```

usd renderHtml

## 1.7

statement.js

```
import createStatementData from "./createStatementData.js";
function statement(invoice, plays) {
  return renderPlainText(createStatementData(invoice, plays));
}
function renderPlainText(data, plays) {
  let result = `Statement for ${data.customer}\n`;
  for (let perf of data.performances) {
    result += ` ${perf.play.name}: ${usd(perf.amount)} (${
      perf.audience
    } seats)\n`;
  }
  result += `Amount owed is ${usd(data.totalAmount)}\n`;
  result += `You earned ${data.totalVolumeCredits} credits\n`;
  return result;
}
function htmlStatement(invoice, plays) {
  return renderHtml(createStatementData(invoice, plays));
}
function renderHtml(data) {
  let result = `<h1>Statement for ${data.customer}</h1>\n`;
  result += "<table>\n";
  result +=
    "<tr><th>play</th><th>seats</th><th>cost</th></tr>";
  for (let perf of data.performances) {
    result += ` <tr><td>${perf.play.name}</td><td>${perf.audience}</td>`;
    result += `<td>${usd(perf.amount)}</td></tr>\n`;
  }
  result += "</table>\n";
  result += `<p>Amount owed is <em>${usd(
    data.totalAmount
  )}</em></p>\n`;
  result += `<p>You earned <em>${data.totalVolumeCredits}</em> credits</p>\n`;
  return result;
}
function usd(aNumber) {
  return new Intl.NumberFormat("en-US", {
    style: "currency",
    currency: "USD",
    minimumFractionDigits: 2,
  }).format(aNumber / 100);
}
```

createStatementData.js

```
export default function createStatementData(invoice, plays) {
const result = {};
result.customer = invoice.customer;
result.performances = invoice.performances.map(enrichPerformance);
result.totalAmount = totalAmount(result);
result.totalVolumeCredits = totalVolumeCredits(result);
return result;

function enrichPerformance(aPerformance) {
  const result = Object.assign({}, aPerformance);
  result.play = playFor(result);
  result.amount = amountFor(result);
  result.volumeCredits = volumeCreditsFor(result);
  return result;
}
function playFor(aPerformance) {
  return plays[aPerformance.playID]
}
function amountFor(aPerformance) {
  let result = 0;
  switch (aPerformance.play.type) {
  case "tragedy":
    result = 40000;
    if (aPerformance.audience > 30) {
      result += 1000 * (aPerformance.audience - 30);
    }
    break;
  case "comedy":
    result = 30000;
    if (aPerformance.audience > 20) {
      result += 10000 + 500 * (aPerformance.audience - 20);
    }
    result += 300 * aPerformance.audience;
    break;
  default:
      throw new Error(`unknown type: ${aPerformance.play.type}`);
  }
  return result;
}
function volumeCreditsFor(aPerformance) {
  let result = 0;
  result += Math.max(aPerformance.audience - 30, 0);
  if ("comedy" === aPerformance.play.type) result += Math.floor(aPerformance
  return result;
}
function totalAmount(data) {
  return data.performances
    .reduce((total, p) => total + p.amount, 0);
}
function totalVolumeCredits(data) {
  return data.performances
```

```
    .reduce((total, p) => total + p.volumeCredits, 0);
}
```

44 70 htmlStatement

HTML

> **Tip**

——

# 1.8

amountFor

——

“ ” —— JavaScript

ECMAScript 2015 ——

“ ” comedy “ ” tragedy amount

volumeCredits 272

272

## createStatementData.js...

```
export default function createStatementData(invoice, plays) {
  const result = {};
  result.customer = invoice.customer;
  result.performances = invoice.performances.map(enrichPerformance);
  result.totalAmount = totalAmount(result);
  result.totalVolumeCredits = totalVolumeCredits(result);
  return result;

  function enrichPerformance(aPerformance) {
    const result = Object.assign({}, aPerformance);
    result.play = playFor(result);
    result.amount = amountFor(result);
    result.volumeCredits = volumeCreditsFor(result);
    return result;
  }

  function playFor(aPerformance) {
    return plays[aPerformance.playID]
  }

  function amountFor(aPerformance) {
    let result = 0;
    switch (aPerformance.play.type) {
      case "tragedy":
        result = 40000;
        if (aPerformance.audience > 30) {
          result += 1000 * (aPerformance.audience - 30);
        }
        break;
      case "comedy":
        result = 30000;
        if (aPerformance.audience > 20) {
          result += 10000 + 500 * (aPerformance.audience - 20);
        }
        result += 300 * aPerformance.audience;
        break;
      default:
        throw new Error(`unknown type: ${aPerformance.play.type}`);
    }
    return result;
  }

  function volumeCreditsFor(aPerformance) {
    let result = 0;
    result += Math.max(aPerformance.audience - 30, 0);
    if ("comedy" === aPerformance.play.type) result += Math.floor(aPerformance
    return result;
  }

  function totalAmount(data) {
    return data.performances
```

```
      .reduce((total, p) => total + p.amount, 0);
  }

  function totalVolumeCredits(data) {
    return data.performances
      .reduce((total, p) => total + p.volumeCredits, 0);
  }
```

enrichPerformance

performance calculator

## function createStatementData...

```
function enrichPerformance(aPerformance) {
  const calculator = new PerformanceCalculator(aPerformance);
  const result = Object.assign({}, aPerformance);
  result.play = playFor(result);
  result.amount = amountFor(result);
  result.volumeCredits = volumeCreditsFor(result);
  return result;
}
```

...

```
class PerformanceCalculator {
  constructor(aPerformance) {
    this.performance = aPerformance;
  }
}
```

——play

124 performance play

## function createStatementData...

```
function enrichPerformance(aPerformance) {
  const calculator = new PerformanceCalculator(
    aPerformance,
    playFor(aPerformance)
  );
  const result = Object.assign({}, aPerformance);
  result.play = calculator.play;
  result.amount = amountFor(result);
  result.volumeCredits = volumeCreditsFor(result);
  return result;
}
```

## class PerformanceCalculator...

```
class PerformanceCalculator {
  constructor(aPerformance, aPlay) {
    this.performance = aPerformance;
    this.play = aPlay;
  }
}
```

“ ”

amount

198 amount PerformanceCalculator aPerformance this.performance playFor(aPerformance) this.play

## class PerformanceCalculator...

```
  get amount() {
  let result = 0;
  switch (this.play.type) {
    case "tragedy":
      result = 40000;
      if (this.performance.audience > 30) {
        result += 1000 * (this.performance.audience - 30);
      }
      break;
    case "comedy":
      result = 30000;
      if (this.performance.audience > 20) {
        result += 10000 + 500 * (this.performance.audience - 20);
      }
      result += 300 * this.performance.audience;
      break;
    default:
      throw new Error(`unknown type: ${this.play.type}`);
  }
  return result;
}
```

Babel

## function createStatementData...

```
function amountFor(aPerformance) {
  return new PerformanceCalculator(aPerformance, playFor(aPerformance)).amount
}
```

115

## function createStatementData...

```
function enrichPerformance(aPerformance) {
  const calculator = new PerformanceCalculator(
    aPerformance,
    playFor(aPerformance)
  );
  const result = Object.assign({}, aPerformance);
  result.play = calculator.play;
  result.amount = calculator.amount;
  result.volumeCredits = volumeCreditsFor(result);
  return result;
}
```

## function createStatementData...

```
function enrichPerformance(aPerformance) {
  const calculator = new PerformanceCalculator(
    aPerformance,
    playFor(aPerformance)
  );
  const result = Object.assign({}, aPerformance);
  result.play = calculator.play;
  result.amount = calculator.amount;
  result.volumeCredits = calculator.volumeCredits;
  return result;
}
```

## class PerformanceCalculator...

```
  get volumeCredits() {
  let result = 0;
  result += Math.max(this.performance.audience - 30, 0);
  if ("comedy" === this.play.type) result += Math.floor(this.performance.audier
  return result;
}
```

362

createStatementData JavaScript

334

## function createStatementData...

```
function enrichPerformance(aPerformance) {
  const calculator = createPerformanceCalculator(
    aPerformance,
    playFor(aPerformance)
  );
  const result = Object.assign({}, aPerformance);
  result.play = calculator.play;
  result.amount = calculator.amount;
  result.volumeCredits = calculator.volumeCredits;
  return result;
}
```

...

```
function createPerformanceCalculator(aPerformance, aPlay) {
  return new PerformanceCalculator(aPerformance, aPlay);
}
```

...

```
function createPerformanceCalculator(aPerformance, aPlay) {
  switch (aPlay.type) {
    case "tragedy":
      return new TragedyCalculator(aPerformance, aPlay);
    case "comedy":
      return new ComedyCalculator(aPerformance, aPlay);
    default:
      throw new Error(`unknown type: ${aPlay.type}`);
  }
}

class TragedyCalculator extends PerformanceCalculator {}
class ComedyCalculator extends PerformanceCalculator {}
```

272

## class TragedyCalculator...

```
get amount() {
  let result = 40000;
  if (this.performance.audience > 30) {
    result += 1000 * (this.performance.audience - 30);
  }
  return result;
}
```

## class PerformanceCalculator...

```
get amount() {
  let result = 0;
  switch (this.play.type) {
    case "tragedy":
      throw 'bad thing';
    case "comedy":
      result = 30000;
      if (this.performance.audience > 20) {
        result += 10000 + 500 * (this.performance.audience - 20);
      }
      result += 300 * this.performance.audience;
      break;
    default:
      throw new Error(`unknown type: ${this.play.type}`);
  }
  return result;
}
```

---

## class ComedyCalculator...

```
get amount() {
  let result = 30000;
  if (this.performance.audience > 20) {
    result += 10000 + 500 * (this.performance.audience - 20);
  }
  result += 300 * this.performance.audience;
  return result;
}
```

amount

## class PerformanceCalculator...

```
get amount() {
  throw new Error('subclass responsibility');
}
```

30

## class PerformanceCalculator...

```
get volumeCredits() {
  return Math.max(this.performance.audience - 30, 0);
}
```

### class ComedyCalculator...

```
get volumeCredits() {
  return super.volumeCredits + Math.floor(this.performance.audience / 5);
}
```

## 1.9

createStatementData.js

```
  export default function createStatementData(invoice, plays) {
  const result = {};
  result.customer = invoice.customer;
  result.performances = invoice.performances.map(enrichPerformance);
  result.totalAmount = totalAmount(result);
  result.totalVolumeCredits = totalVolumeCredits(result);
  return result;

  function enrichPerformance(aPerformance) {
    const calculator = createPerformanceCalculator(aPerformance, playFor(aPerf
    const result = Object.assign({}, aPerformance);
    result.play = calculator.play;
    result.amount = calculator.amount;
    result.volumeCredits = calculator.volumeCredits;
    return result;
  }
  function playFor(aPerformance) {
    return plays[aPerformance.playID]
  }
  function totalAmount(data) {
    return data.performances
      .reduce((total, p) => total + p.amount, 0);
  }
  function totalVolumeCredits(data) {
    return data.performances
      .reduce((total, p) => total + p.volumeCredits, 0);
  }
}
function createPerformanceCalculator(aPerformance, aPlay) {
    switch(aPlay.type) {
    case "tragedy": return new TragedyCalculator(aPerformance, aPlay);
    case "comedy" : return new ComedyCalculator(aPerformance, aPlay);
    default:
        throw new Error(`unknown type: ${aPlay.type}`);
    }
}
class PerformanceCalculator {
  constructor(aPerformance, aPlay) {
    this.performance = aPerformance;
    this.play = aPlay;
  }
  get amount() {
    throw new Error('subclass responsibility');
  }
  get volumeCredits() {
    return Math.max(this.performance.audience - 30, 0);
  }
}
class TragedyCalculator extends PerformanceCalculator {
  get amount() {
    let result = 40000;
```

```
    if (this.performance.audience > 30) {
      result += 1000 * (this.performance.audience - 30);
    }
    return result;
  }
}
class ComedyCalculator extends PerformanceCalculator {
  get amount() {
    let result = 30000;
    if (this.performance.audience > 20) {
      result += 10000 + 500 * (this.performance.audience - 20);
    }
    result += 300 * this.performance.audience;
    return result;
  }
  get volumeCredits() {
    return super.volumeCredits + Math.floor(this.performance.audience / 5);
  }
}
```

amountFor volumeCreditsFor

createPerformanceCalculator

createStatementData JavaScript

“ ” “ ”

# 1.10

“ ” 106 123 198 272

3 154

> **Tip**

20 Kent Beck

# 2

## 2.1

“ ” ——— 1
“ ”

106 272

“ ”

> **Tip**

“ ” restructuring
——— ———

“ ”
106 124 198
bug bug bug

“ ”

## 2.2

Kent Beck “ ”

…… 10

## 2.3

“ ” “ ” “ ”

——

“ ” “ ”

“ ”

——

——

Maudite

## bug

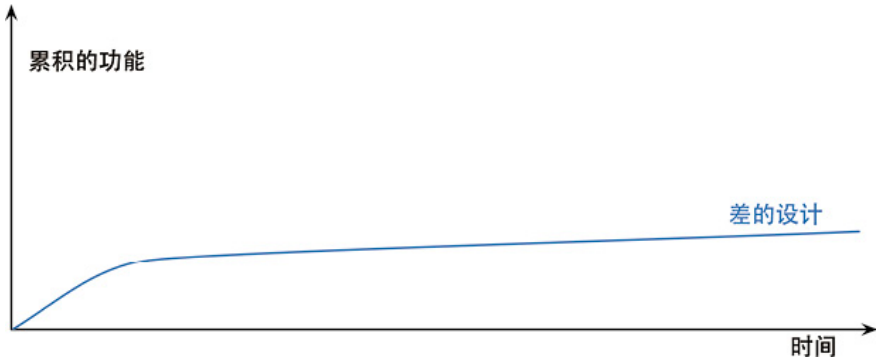
bug bug bug

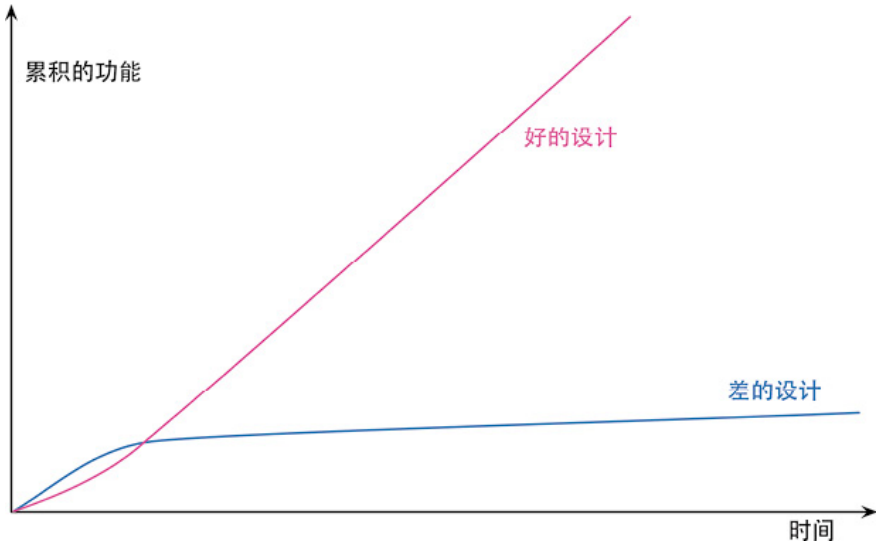
bug

Kent Beck “ ”

bug

bug






bug bug

“ ” “ ”

20

## 2.4

> **Tip**

Don Roberts

——

310

> **Tip**
>
> 100 20 100 3 “
> ” “ ”

——Jessica Kerr

bug 3

bug

“ ”

Ward Cunningham

——

Ralph Johnson

——

—— —— bug

bug if

> **Tip**

> **Tip**

——Kent Beck

“

”

“ ”

—— ——

“ ”

——

Branch By Abstraction[mf-bba]

code review

pull request

“ ?” “ ” ____

____

“ ” “ ”

API

## 2.5

“ ”

> **Tip**

____

——

——

“ ”

“ ”“ ”

—— bug “ ”

124

API

published interface

124

“ ” deprecated

B A A

“ ” “ ”

master trunk

integrate merge

rebase ——

“ ” pull push Rachel

Rachel

4 2

Continuous Integration CI “

” Trunk-Based Development CI CI

feature toggle feature flag

CI CI

CI Kent Beck

CI

CI CI

[Forsgren et al]

“ ”

——

“ ”

[mf-stc]

“ ” 20

bug

bug

“ bug”

Jay Bazuzi C++ 106 [Bazuzi]

——

“ ” ——

[Feathers]

——

——

1 “ ” 1 Pramod Sadalage

[mf-evodb] [Ambler & Sadalage]

bug bug

Parallel Change /expand-contract [mf-pc]

## 2.6 YAGNI

“ ”

——

“ ”

——

310 “ ”

YAGNI[mf-yagni]——“ ” you aren’t going to need it

YAGNI “ ” YAGNI

YAGNI

[Ford et al.]

## 2.7

“ ” XP [mf-xp] XP

[mf-nm] “ ”

CI

—— ——

YAGNI YAGNI YAGNI YAGNI

“ ”

bug

——

——

# 2.8

“ ”

3

> **Tip**

Kent Beck Martin Fowler

“ ”

Kent “ ” instance

“ ” ——

“ ”

“ ”

“ ”

Kent Martin

——Ron Jeffries

90

90% ——

“ ”

# 2.9

“ ” refactoring

Ward Cunningham Kent Beck 20 80
Smalltalk Smalltalk Smalltalk “ - - ”
—— Ward Kent
Kent

Ward Kent Smalltalk Smalltalk Smalltalk Ralph
Johnson - GoF[gof] Ralph

Bill Opdyke Ralph Smalltalk
Bill Bill C++
“ ” semantics-preserving refactoring Bill
[Opdyke]

1992 OOPSLA Bill Bill “ ”

John Brant Don Roberts “ ” Refactoring Browser
Smalltalk

Kent

1

1 “ ” Java
“ ”

# 2.10

10 Java IntelliJ IDEA Eclipse

Smalltalk Refactoring Browser John Brandt Don Roberts 21 Java
JetBrains IntelliJ IDEA IDE IBM
VisualAge Java VisualAge Eclipse

C# JetBrains Resharper Visual Studio Visual Studio IDE

/ 119

Emacs

IDE IDE

IDE

124 Salesman Server addClient

Salesman addClient Server addClient

Smalltalk Refactoring Browser

Java 124

106

IDE Emacs Java IntelliJ IDEA

Eclipse

Java Method.invoke

IDE Language

Server API

# 2.11

2

1 Bill Wake [Wake]

Josh Kerievsky [Kerievsky] GoF[gof]

[Ambler & Sadalage] Scott Ambler Pramod

Sadalage HTML [Harold] Elliotte Rusty Harold

“ ” Michael Feathers [Feathers]

Jay Fields Shane Harvey Ruby

[Fields et al.]

Web refactoring.com [ref.com]

# 3

——Kent Beck Martin Fowler

“ ”

—— Beck

“ ” “ ”

“ ” “ ” “ ”

1 Kent Beck

“ ” “ ”

“ ” Kent ——

“ ”

“ ” “ ”

“ ”

## 3.1 Mysterious Name

1

[mf-2h] 124 137 244

1 International Man of Mystery 1997 · ——

## 3.2 Duplicated Code

“ ” 106

223 350

## 3.3 Long Function

“ ”——

—— ——

“ ” “ ”

106

106

178 140 319

—— 337

260 switch 106

switch 272

227

## 3.4 Long Parameter List

324 319

140 flag 314

144

partially applied function

## 3.5 Global Data

——

bug

singleton

132

## 3.6 Mutable Data

bug ——

—— —— “ ”

——

132 240

223 106 API

306 331 ——

bug 248

144 149

252

## 3.7 Divergent Change

—— “ ”

“ 3 4 ”

“

”

154

198 106

182

## 3.8 Shotgun Surgery

198 207 144

149 154

inline —— 115 186 ——

## 3.9 Feature Envy

198

106 198

106

GoF[gof] Strategy Visitor Kent Beck Self Delegation [Beck SBPP]

—— “ ”

## 3.10 Data Clumps

182 140 319

“ ”

## 3.11 Primitive Obsession

if (a < upper && a > lower)

“ ”

“ ” stringly typed

174
362 272

182 140

## 3.12 switch Repeated Switches

switch switch 272
if

1 “switch ” Switch Statements 20 90

switch 15 switch
switch switch switch/case if/else switch
switch

## 3.13 Loops

1
Java 231 filter
map

## 3.14 Lazy Element

115 186 380

## 3.15 Speculative Generality

Brian Foote “ ”

380 115 186 124
124

“ ” 237

## 3.16 Temporary Field

182 198 289 “ ”

## 3.17 Message Chains

……

189 “ ”
106 198

## 3.18 Middle Man

—— “ ”

192 “ ”
115 399 381

## 3.19 Insider Trading

198 207
189

381 399

## 3.20 Large Class

182 depositAmount depositCurrency
375 362

“ ” 5 “ ”
5 “ ” 10 “ ”

182
375 362

## 3.21 Alternative Classes with Different Interfaces

124
198 375

## 3.22 Data Class

public
162 331
/ 198 106

154 154

## 3.23 Refused Bequest

359 361
abstract
“ ”
“ ”
“ ”
381 399

## 3.24 Comments

“ ”

106 124

302

> **Tip**

“ ”

# 4

## 4.1

—— bug bug
bug bug bug

“ ” 1992 OOPSLA Bedarra Dave Thomas “
”

——

“OK” “ ”

bug bug bug
bug bug bug
bug

> **Tip**

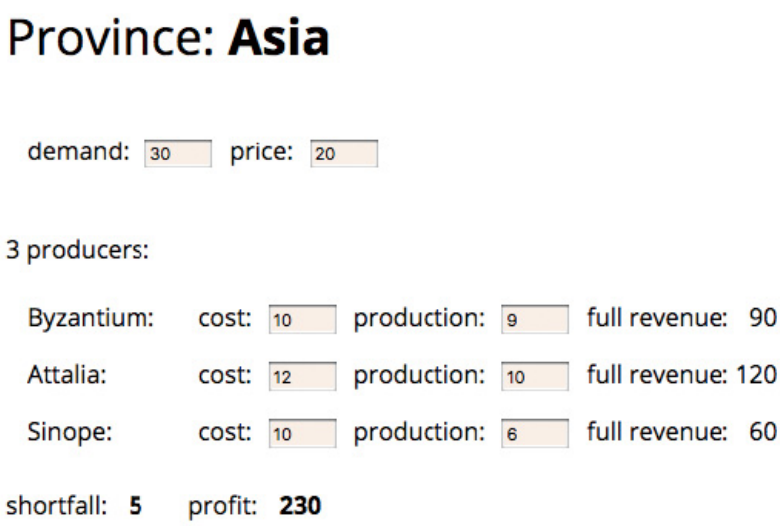
1997 Kent Beck OOPSLA
Erich Gamma Smalltalk Java JUnit
[mf-xunit]

> **Tip**
> bug bug

Kent Beck Test-Driven Development TDD [mf-tdd]
“ ”

JavaScript

## 4.2




province demand price producer cost full revenue profit production

HTML UI

Producer Province Province JavaScript JSON

JSON

## class Province...

```
constructor(doc) {
 this._name = doc.name;
 this._producers = [];
 this._totalProduction = 0;
 this._demand = doc.demand;
 this._price = doc.price;
 doc.producers.forEach(d => this.addProducer(new Producer(this, d)));
}
addProducer(arg) {
 this._producers.push(arg);
 this._totalProduction += arg.production;
}
```

JSON

...

```
function sampleProvinceData() {
  return {
    name: "Asia",
    producers: [
      { name: "Byzantium", cost: 10, production: 9 },
      { name: "Attalia", cost: 12, production: 10 },
      { name: "Sinope", cost: 10, production: 6 },
    ],
    demand: 30,
    price: 20,
  };
}
```

## class Province...

```
get name()    {return this._name;}
get producers() {return this._producers.slice();}
get totalProduction()    {return this._totalProduction;}
set totalProduction(arg) {this._totalProduction = arg;}
get demand()    {return this._demand;}
set demand(arg) {this._demand = parseInt(arg);}
get price()    {return this._price;}
set price(arg) {this._price = parseInt(arg);}
```

UI

Producer

## class Producer...

```
constructor(aProvince, data) {
 this._province = aProvince;
 this._cost = data.cost;
 this._name = data.name;
 this._production = data.production || 0;
}
get name() {return this._name;}
get cost() {return this._cost;}
set cost(arg) {this._cost = parseInt(arg);}

get production() {return this._production;}
set production(amountStr) {
 const amount = parseInt(amountStr);
 const newProduction = Number.isNaN(amount) ? 0 : amount;
 this._province.totalProduction += newProduction - this._production;
 this._production = newProduction;
}
```

production

## class Province...

```
get shortfall() {
 return this._demand - this.totalProduction;
}
```

## class Province...

```
get profit() {
 return this.demandValue - this.demandCost;
}
get demandCost() {
 let remainingDemand = this.demand;
 let result = 0;
 this.producers
  .sort((a,b) => a.cost - b.cost)
  .forEach(p => {
   const contribution = Math.min(remainingDemand, p.production);
    remainingDemand -= contribution;
    result += contribution * p.cost;
  });
 return result;
}
get demandValue() {
 return this.satisfiedDemand * this.price;
}
get satisfiedDemand() {
 return Math.min(this._demand, this.totalProduction);
}
```

# 4.3

JavaScript Mocha

```
describe("province", function () {
  it("shortfall", function () {
    const asia = new Province(sampleProvinceData());
    assert.equal(asia.shortfall, 5);
  });
});
```

Mocha it fixture

> **Tip**
>
> describe it

NodeJS

```
''''''''''''''

1 passing (61ms)
```

bug

> **Tip**

## class Province...

```
get shortfall() {
 return this._demand - this.totalProduction * 2;
}
```

```
!

0 passing (72ms)
1 failing

1) province shortfall:
AssertionError: expected -20 to equal 5
at Context.<anonymous> (src/tester.js:10:12)
```

———

> **Tip**

Mocha JavaScript
Chai “assert”

```
describe("province", function () {
  it("shortfall", function () {
    const asia = new Province(sampleProvinceData());
    assert.equal(asia.shortfall, 5);
  });
});
```

“expect”

```
describe("province", function () {
  it("shortfall", function () {
    const asia = new Province(sampleProvinceData());
    expect(asia.shortfall).equal(5);
  });
});
```

assert JavaScript expect

Java IDE

“ ” “ ” “ ” “ ”

Emacs

## 4.4

“ public

” bug

> **Tip**

```
describe("province", function () {
  it("shortfall", function () {
    const asia = new Province(sampleProvinceData());
    expect(asia.shortfall).equal(5);
  });
  it("profit", function () {
    const asia = new Province(sampleProvinceData());
    expect(asia.profit).equal(230);
  });
});
```

230

2

——

```
describe("province", function () {
  const asia = new Province(sampleProvinceData()); // DON'T DO THIS
  it("shortfall", function () {
    expect(asia.shortfall).equal(5);
  });
  it("profit", function () {
    expect(asia.profit).equal(230);
  });
});
```

bug —— bug

JavaScript const asia

```
describe("province", function () {
  let asia;
  beforeEach(function () {
    asia = new Province(sampleProvinceData());
  });
  it("shortfall", function () {
    expect(asia.shortfall).equal(5);
  });
  it("profit", function () {
    expect(asia.profit).equal(230);
  });
});
```

beforeEach asia

beforeEach it

beforeEach describe

# 4.5

bug Producer production

```
describe('province'...
  it('change production', function() {
  asia.producers[0].production = 20;
  expect(asia.shortfall).equal(-6);
  expect(asia.profit).equal(292);
});
```

beforeEach

- - setup-exercise-verify given-when-then - - arrange-act-assert

beforeEach

> **Tip**
>
> beforeEach

it it

it

# 4.6

“ ” happy path

```
describe('no producers', function() {
 let noProducers;
 beforeEach(function() {
  const data = {
   name: "No proudcers",
   producers: [],
   demand: 30,
   price: 20
  };
  noProducers = new Province(data);
 });
 it('shortfall', function() {
  expect(noProducers.shortfall).equal(30);
 });
 it('profit', function() {
  expect(noProducers.profit).equal(0);
 });
```

0

```
describe('province'...
  it('zero demand', function() {
 asia.demand = 0;
  expect(asia.shortfall).equal(-25);
  expect(asia.profit).equal(0);
 });
```

```
describe('province'...
  it('negative demand', function() {
 asia.demand = -1;
 expect(asia.shortfall).equal(-26);
 expect(asia.profit).equal(-10);
});
```

0

0

UI

> **Tip**

```
describe('province'...
  it('empty string demand', function() {
 asia.demand = "";
 expect(asia.shortfall).NaN;
 expect(asia.profit).NaN;
});
```

“ ” ——

```
describe('string for producers', function() {
 it('', function() {
  const data = {
   name: "String producers",
   producers: "",
   demand: 30,
   price: 20
  };
  const prov = new Province(data);
  expect(prov.shortfall).equal(0);
 });
```

0

```
''''''''!

9 passing (74ms)
1 failing

1) string for producers :
  TypeError: doc.producers.forEach is not a function
  at new Province (src/main.js:22:19)
  at Context.<anonymous> (src/tester.js:86:18)
```

Mocha failure error “ ”
“ ” JavaScript
“...is not a function”

producers

JSON

> **Tip**
>
> 302

“ bug ” “ ”

bug bug

> **Tip**
>
> bug bug

# 4.7

1

bug

bug bug

“ ” [mf-tc]

> **Tip**
>
> bug bug

bug

——

# 5

## 5.1

5

- name
- sketch
- motivation “ ” “ ”
- mechanics
- examples

“ ” “ ” “ ” “ ”
“ ” “ ” “ ”

“ ” “ ”

“ ”

## 5.2

“ ”

1 223 2

132

# 6

106 119 ——
115 123

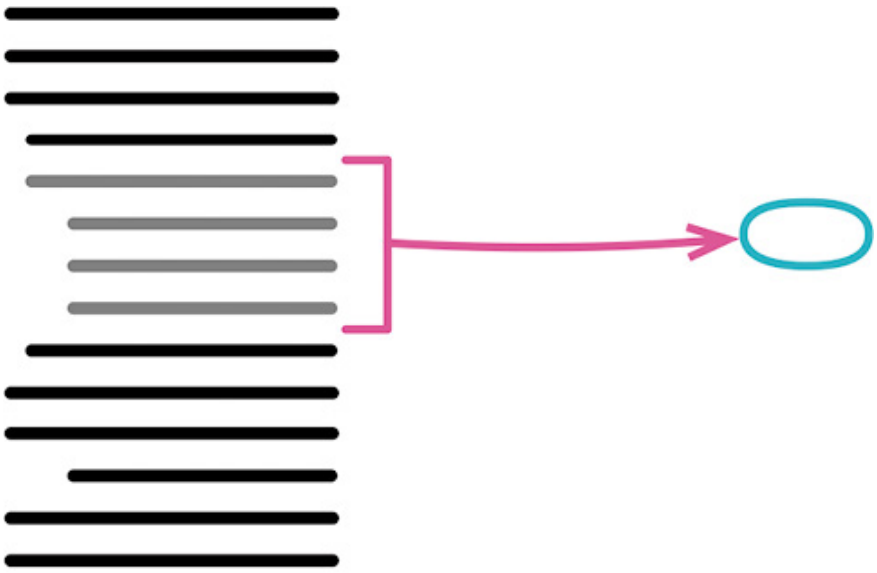
124 137
132 140

144
149 transform 154

## 6.1 Extract Function

Extract Method

115



```
function printOwing(invoice) {
  printBanner();
  let outstanding = calculateOutstanding();

  //print details
  console.log(`name: ${invoice.customer}`);
  console.log(`amount: ${outstanding}`);
}

function printOwing(invoice) {
  printBanner();
  let outstanding = calculateOutstanding();
  printDetails(outstanding);

  function printDetails(outstanding) {
    console.log(`name: ${invoice.customer}`);
    console.log(`amount: ${outstanding}`);
  }
}
```

“ /function” “ /method” “ /procedure” “ /subroutine”

“ ”

inline “ ”

—— 6 1 Kent Beck Smalltalk “ ” Smalltalk Smalltalk highlight reverse highlight

“ ”

- “ ” “ ”

> **Tip**

198

- 
- 

> **Tip**

“ ”

query

240 178

- 

> **Tip**

- 
- 
- 222

> **Tip**

```
function printOwing(invoice) {
  let outstanding = 0;

  console.log("************************");
  console.log("**** Customer Owes ****");
  console.log("************************");

  // calculate outstanding
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  // record due date
  const today = Clock.today;
  invoice.dueDate = new Date(
    today.getFullYear(),
    today.getMonth(),
    today.getDate() + 30
  );

  //print details
  console.log(`name: ${invoice.customer}`);
  console.log(`amount: ${outstanding}`);
  console.log(`due: ${invoice.dueDate.toLocaleDateString()}`);
}
```

Clock.today Clock Wrapper[mf-cw] Date.now()

“ ”

```
function printOwing(invoice) {
  let outstanding = 0;

  printBanner();

  // calculate outstanding
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  // record due date
  const today = Clock.today;
  invoice.dueDate = new Date(
    today.getFullYear(),
    today.getMonth(),
    today.getDate() + 30
  );

  //print details
  console.log(`name: ${invoice.customer}`);
  console.log(`amount: ${outstanding}`);
  console.log(`due: ${invoice.dueDate.toLocaleDateString()}`);
}
function printBanner() {
  console.log("************************");
  console.log("**** Customer Owes ****");
  console.log("************************");
}
```

“ ”

```
function printOwing(invoice) {
 let outstanding = 0;

 printBanner();

 // calculate outstanding
 for (const o of invoice.orders) {
  outstanding += o.amount;
 }

 // record due date
 const today = Clock.today;
 invoice.dueDate = new Date(today.getFullYear(), today.getMonth(), today.getDat

 printDetails();

 function printDetails() {
  console.log(`name: ${invoice.customer}`);
  console.log(`amount: ${outstanding}`);
  console.log(`due: ${invoice.dueDate.toLocaleDateString()}`);
}
```

printDetails printOwing printOwing
“ ” “ ”

```
function printOwing(invoice) {
 let outstanding = 0;

 printBanner();

 // calculate outstanding
 for (const o of invoice.orders) {
  outstanding += o.amount;
 }

 // record due date
 const today = Clock.today;
 invoice.dueDate = new Date(today.getFullYear(), today.getMonth(), today.getDat

 //print details
 console.log(`name: ${invoice.customer}`);
 console.log(`amount: ${outstanding}`);
 console.log(`due: ${invoice.dueDate.toLocaleDateString()}`);
```

“ ”

```
function printOwing(invoice) {
  let outstanding = 0;

  printBanner();

  // calculate outstanding
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  // record due date
  const today = Clock.today;
  invoice.dueDate = new Date(
    today.getFullYear(),
    today.getMonth(),
    today.getDate() + 30
  );

  printDetails(invoice, outstanding);
}
function printDetails(invoice, outstanding) {
  console.log(`name: ${invoice.customer}`);
  console.log(`amount: ${outstanding}`);
  console.log(`due: ${invoice.dueDate.toLocaleDateString()}`);
}
```

“ ”

```
function printOwing(invoice) {
  let outstanding = 0;

  printBanner();

  // calculate outstanding
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
function recordDueDate(invoice) {
  const today = Clock.today;
  invoice.dueDate = new Date(
    today.getFullYear(),
    today.getMonth(),
    today.getDate() + 30
  );
}
```

240

223

```
function printOwing(invoice) {
  let outstanding = 0;

  printBanner();

  // calculate outstanding
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
```

“ ”

```
function printOwing(invoice) {
  printBanner();

  // calculate outstanding
  let outstanding = 0;
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
```

```
function printOwing(invoice) {
  printBanner();

  // calculate outstanding
  let outstanding = 0;
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }

  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
function calculateOutstanding(invoice) {
  let outstanding = 0;
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }
  return outstanding;
}
```

outstanding outstanding

JavaScript —— “ ” “ ”

outstanding

```
function printOwing(invoice) {
  printBanner();
  let outstanding = calculateOutstanding(invoice);
  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
function calculateOutstanding(invoice) {
  let outstanding = 0;
  for (const o of invoice.orders) {
    outstanding += o.amount;
  }
  return outstanding;
}
```

```
function printOwing(invoice) {
  printBanner();
  const outstanding = calculateOutstanding(invoice);
  recordDueDate(invoice);
  printDetails(invoice, outstanding);
}
function calculateOutstanding(invoice) {
  let result = 0;
  for (const o of invoice.orders) {
    result += o.amount;
  }
  return result;
}
```

outstanding const

“ ”

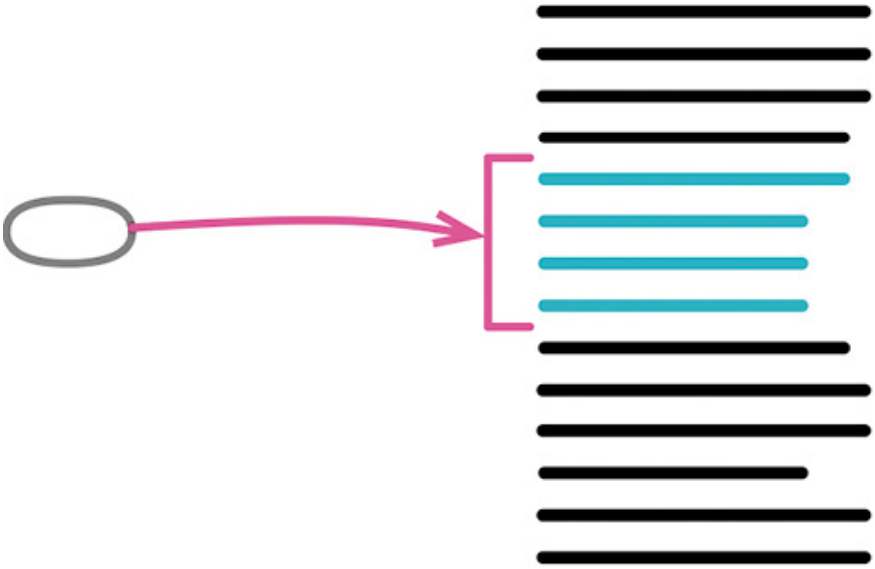
—

178 240

## 6.2 Inline Function

Inline Method

106



```
function getRating(driver) {
 return moreThanFiveLateDeliveries(driver) ? 2 : 1;
}

function moreThanFiveLateDeliveries(driver) {
 return driver.numberOfLateDeliveries > 5;
}


function getRating(driver) {
 return (driver.numberOfLateDeliveries > 5) ? 2 : 1;
}
```

- 

> **Tip**

- 
- 
- 

> **Tip**

-

```
function rating(aDriver) {
  return moreThanFiveLateDeliveries(aDriver) ? 2 : 1;
}
function moreThanFiveLateDeliveries(aDriver) {
  return aDriver.numberOfLateDeliveries & gt;
  5;
}
```

return

```
function rating(aDriver) {
  return aDriver.numberOfLateDeliveries & amp;
  gt;
  5 ? 2 : 1;
}
```

```
function rating(aDriver) {
  return moreThanFiveLateDeliveries(aDriver) ? 2 : 1;
}

function moreThanFiveLateDeliveries(dvr) {
  return dvr.numberOfLateDeliveries & gt;
  5;
}
```

moreThanFiveLateDeliveries

```
function rating(aDriver) {
  return aDriver.numberOfLateDeliveries & gt;
  5 ? 2 : 1;
}
```

```
function reportLines(aCustomer) {
  const lines = [];
  gatherCustomerData(lines, aCustomer);
  return lines;
}
function gatherCustomerData(out, aCustomer) {
  out.push(["name", aCustomer.name]);
  out.push(["location", aCustomer.location]);
}
```

gatherCustomerData reportLines

217 —— “ - - ”

```
function reportLines(aCustomer) {
  const lines = [];
  lines.push(["name", aCustomer.name]);
  gatherCustomerData(lines, aCustomer);
  return lines;
}
function gatherCustomerData(out, aCustomer) {
  out.push(["name", aCustomer.name]);
  out.push(["location", aCustomer.location]);
}
```

```
function reportLines(aCustomer) {
  const lines = [];
  lines.push(["name", aCustomer.name]);
  lines.push(["location", aCustomer.location]);
  return lines;
}
```

217

# 6.3 Extract Variable

Introduce Explaining Variable

123

```
return (
  order.quantity * order.itemPrice -
  Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
  Math.min(order.quantity * order.itemPrice * 0.1, 100)
);

const basePrice = order.quantity * order.itemPrice;
const quantityDiscount =
  Math.max(0, order.quantity - 500) * order.itemPrice * 0.05;
const shipping = Math.min(basePrice * 0.1, 100);
return basePrice - quantityDiscount + shipping;
```

“ ” 178

106

- 
- 
- 
- 

```
function price(order) {
  //price is base price - quantity discount + shipping
  return (
    order.quantity * order.itemPrice -
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
    Math.min(order.quantity * order.itemPrice * 0.1, 100)
  );
}
```

base price quantity item price

```
function price(order) {
  //price is base price - quantity discount + shipping
  return (
    order.quantity * order.itemPrice -
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
    Math.min(order.quantity * order.itemPrice * 0.1, 100)
  );
}
```

```
function price(order) {
  //price is base price - quantity discount + shipping
  const basePrice = order.quantity * order.itemPrice;
  return (
    order.quantity * order.itemPrice -
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
    Math.min(order.quantity * order.itemPrice * 0.1, 100)
  );
}
```

```
function price(order) {
  //price is base price - quantity discount + shipping
  const basePrice = order.quantity * order.itemPrice;
  return (
    basePrice -
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
    Math.min(order.quantity * order.itemPrice * 0.1, 100)
  );
}
```

```
function price(order) {
  //price is base price - quantity discount + shipping
  const basePrice = order.quantity * order.itemPrice;
  return (
    basePrice -
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05 +
    Math.min(basePrice * 0.1, 100)
  );
}
```

quantity discount

```
function price(order) {
  //price is base price - quantity discount + shipping
  const basePrice = order.quantity * order.itemPrice;
  const quantityDiscount =
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05;
  return basePrice - quantityDiscount + Math.min(basePrice * 0.1, 100);
}
```

shipping

```
function price(order) {
  const basePrice = order.quantity * order.itemPrice;
  const quantityDiscount =
    Math.max(0, order.quantity - 500) * order.itemPrice * 0.05;
  const shipping = Math.min(basePrice * 0.1, 100);
  return basePrice - quantityDiscount + shipping;
}
```

```
class Order {
  constructor(aRecord) {
    this._data = aRecord;
  }

  get quantity() {
    return this._data.quantity;
  }
  get itemPrice() {
    return this._data.itemPrice;
  }

  get price() {
    return (
      this.quantity * this.itemPrice -
      Math.max(0, this.quantity - 500) * this.itemPrice * 0.05 +
      Math.min(this.quantity * this.itemPrice * 0.1, 100)
    );
  }
}
```

Order “ ”

```
class Order {
  constructor(aRecord) {
    this._data = aRecord;
  }
  get quantity() {
    return this._data.quantity;
  }
  get itemPrice() {
    return this._data.itemPrice;
  }

  get price() {
    return this.basePrice - this.quantityDiscount + this.shipping;
  }
  get basePrice() {
    return this.quantity * this.itemPrice;
  }
  get quantityDiscount() {
    return Math.max(0, this.quantity - 500) * this.itemPrice * 0.05;
  }
  get shipping() {
    return Math.min(this.basePrice * 0.1, 100);
  }
}
```

# 6.4 Inline Variable

Inline Temp

119

```
let basePrice = anOrder.basePrice;
return (basePrice > 1000);
```

```
return anOrder.basePrice & gt;
1000;
```

- 
- 

> **Tip**

- 
- 
- 
- 
- 

# 6.5 Change Function Declaration

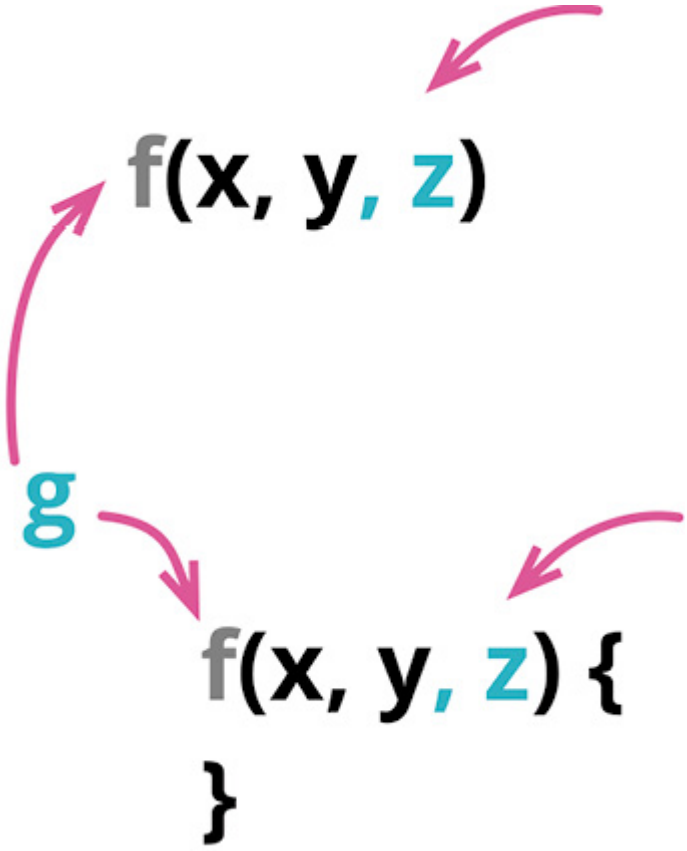
Rename Function

Rename Method

Add Parameter

Remove Parameter

Change Signature




```
function circum(radius) {...}
```

```
function circumference(radius) {...}
```

____

“

____ ”

person company “ ” “ ”

“ ” “ ”

____

—— 30 “ ” payment
“ ” ——

“ ”

“ ” rename Rename
Function

- 
- 106

> **Tip**

- 
- 
- 115
- 124
- 

“ ”

API “ ” deprecated

```
function circum(radius) {
  return 2 * Math.PI * radius;
}
```

```
function circumference(radius) {
  return 2 * Math.PI * radius;
}
```

circum circumference

“ ” IDE

“ ” Person changeAddress “ ” InsuranceAgreement

```
function circum(radius) {
  return 2 * Math.PI * radius;
}
```

106

```
function circum(radius) {
  return circumference(radius);
}
function circumference(radius) {
  return 2 * Math.PI * radius;
}
```

115

API—— API circumference circum deprecated circumference circum “ circum ”

“ ” Book customer reservation

## class Book...

```
  addReservation(customer) {
  this._reservations.push(customer);
}
```

“ ” addReservation

106 addReservation addReservation

## class Book...

```
addReservation(customer) {
  this.zz_addReservation(customer);
}
zz_addReservation(customer) {
  this._reservations.push(customer);
}
```

## class Book...

```
addReservation(customer) {
  this.zz_addReservation(customer, false);
}

zz_addReservation(customer, isPriority) {
  this._reservations.push(customer);
}
```

JavaScript 302

## class Book...

```
zz_addReservation(customer, isPriority) {
  assert(isPriority === true || isPriority === false);
  this._reservations.push(customer);
}
```

——

115

customer New England

```
function inNewEngland(aCustomer) {
  return ["MA", "CT", "ME", "VT", "NH", "RI"].includes(aCustomer.address.state)
}
```

•••

```
  const newEnglanders = someCustomers.filter(c => inNewEngland(c));
```

inNewEngland state state code
“ ”

106 119

```
function inNewEngland(aCustomer) {
  const stateCode = aCustomer.address.state;
  return ["MA", "CT", "ME", "VT", "NH", "RI"].includes(stateCode);
}
```

106

```
function inNewEngland(aCustomer) {
  const stateCode = aCustomer.address.state;
  return xxNEWinNewEngland(stateCode);
}

function xxNEWinNewEngland(stateCode) {
  return ["MA", "CT", "ME", "VT", "NH", "RI"].includes(stateCode);
}
```

123

```
function inNewEngland(aCustomer) {
  return xxNEWinNewEngland(aCustomer.address.state);
}
```

115

•••

```
const newEnglanders = someCustomers.filter(c => xxNEWinNewEngland(c.address.s
```

...

```
const newEnglanders = someCustomers.filter(c => inNewEngland(c.address.state)
```

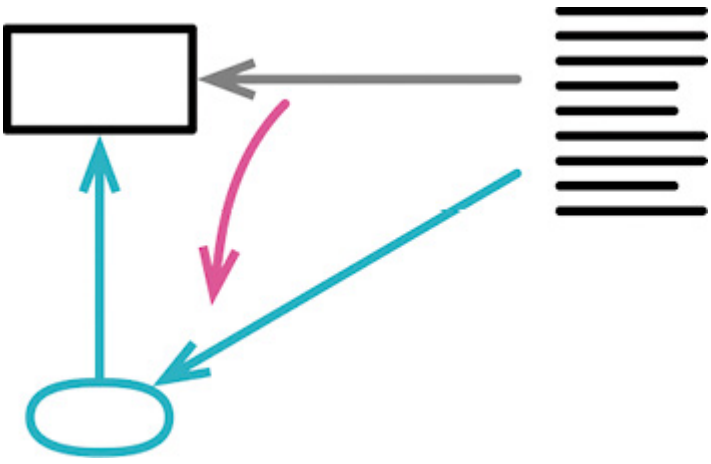
...

```
function inNewEngland(stateCode) {
  return ["MA", "CT", "ME", "VT", "NH", "RI"].includes(stateCode);
}
```

# 6.6 Encapsulate Variable

Self-Encapsulate Field

Encapsulate Field



```
let defaultOwner = { firstName: "Martin", lastName: "Fowler" };
```

```
let defaultOwnerData = { firstName: "Martin", lastName: "Fowler" };
export function defaultOwner() {
  return defaultOwnerData;
}
export function setDefaultOwner(arg) {
  defaultOwnerData = arg;
}
```

“ ” “ ”

private public

____ “ ” ____

____

- 
- 
- 
- 

> **Tip**

- 
- 162

```
let defaultOwner = { firstName: "Martin", lastName: "Fowler" };
```

```
spaceship.owner = defaultOwner;
```

```
defaultOwner = { firstName: "Rebecca", lastName: "Parsons" };
```

```
function getDefaultOwner() {
  return defaultOwner;
}
function setDefaultOwner(arg) {
  defaultOwner = arg;
}
```

defaultOwner

```
spaceship.owner = getDefaultOwner();
```

```
setDefaultOwner({ firstName: "Rebecca", lastName: "Parsons" });
```

JavaScript

## defaultOwner.js...

```
let defaultOwner = { firstName: "Martin", lastName: "Fowler" };
export function getDefaultOwner() {
  return defaultOwner;
}
export function setDefaultOwner(arg) {
  defaultOwner = arg;
}
```

`__privateOnly_defaultOwner`

get

## defaultOwner.js...

```
let defaultOwnerData = { firstName: "Martin", lastName: "Fowler" };
export function getdefaultOwner() {
  return defaultOwnerData;
}
export function setDefaultOwner(arg) {
  defaultOwnerData = arg;
}
```

JavaScript “ / ” Overloaded Getter Setter [mf-orgs] get set

```
const owner1 = defaultOwner();
assert.equal("Fowler", owner1.lastName, "when set");
const owner2 = defaultOwner();
owner2.lastName = "Parsons";
assert.equal("Parsons", owner1.lastName, "after change owner2"); // is this ok?
```

## defaultOwner.js...

```
let defaultOwnerData = { firstName: "Martin", lastName: "Fowler" };
export function defaultOwner() {
  return Object.assign({}, defaultOwnerData);
}
export function setDefaultOwner(arg) {
  defaultOwnerData = arg;
}
```

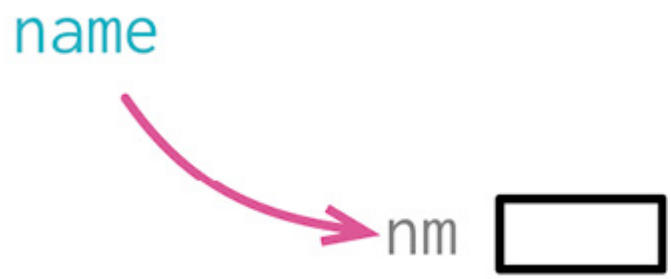
```
  let defaultOwnerData = {firstName: "Martin", lastName: "Fowler"};
export function defaultOwner()        {return new Person(defaultOwnerData);}
export function setDefaultOwner(arg) {defaultOwnerData = arg;}

class Person {
 constructor(data) {
  this._lastName = data.lastName;
  this._firstName = data.firstName
 }
 get lastName() {return this._lastName;}
 get firstName() {return this._firstName;}
 // and so on for other properties
```

defaultOwner “ ” lastName firstName

“ ”

“ ”

## 6.7 Rename Variable




```
let a = height * width;
```

```
let area = height * width;
```

—— ——

lambda ——

JavaScript aCustomer

- 132
-

> **Tip**
>
> “ ” published variable

-

```
let tpHd = "untitled";
```

```
result += `<h1>${tpHd}</h1>`;
```

```
tpHd = obj["articleTitle"];
```

132

```
result += `<h1>${title()}</h1>`;

setTitle(obj["articleTitle"]);

function title() {
  return tpHd;
}
function setTitle(arg) {
  tpHd = arg;
}
```

```
let _title = "untitled";

function title() {
  return _title;
}
function setTitle(arg) {
  _title = arg;
}
```

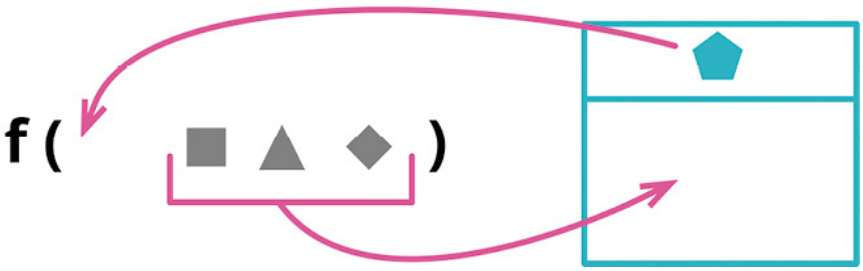
> **Tip**
>
> getTitle

```
const cpyNm = "Acme Gooseberries";
```

```
const companyName = "Acme Gooseberries";
const cpyNm = companyName;
```

JavaScript

## 6.8 Introduce Parameter Object




```
function amountInvoiced(startDate, endDate) {...}
function amountReceived(startDate, endDate) {...}
function amountOverdue(startDate, endDate) {...}
```

```
function amountInvoiced(aDateRange) {...}
function amountReceived(aDateRange) {...}
function amountOverdue(aDateRange) {...}
```

—— ——

- 

> **Tip**
>
> [mf-vo]

- 
- 124
- 
- 
- 

reading range

```
const station = {
  name: "ZB1",
  readings: [
    { temp: 47, time: "2016-11-10 09:10" },
    { temp: 53, time: "2016-11-10 09:20" },
    { temp: 58, time: "2016-11-10 09:30" },
    { temp: 53, time: "2016-11-10 09:40" },
    { temp: 51, time: "2016-11-10 09:50" },
  ],
};
```

```
function readingsOutsideRange(station, min, max) {
 return station.readings
  .filter(r => r.temp < min || r.temp > max);
}
```

```
alerts = readingsOutsideRange(
  station,
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling
);
```

readingsOutsideRange “ ” operatingPlan

readingsOutsideRange

```
class NumberRange {
  constructor(min, max) {
    this._data = { min: min, max: max };
  }
  get min() {
    return this._data.min;
  }
  get max() {
    return this._data.max;
  }
}
```

JavaScript

Value Object [mf-vo]

124 readingsOutsideRange

```
function readingsOutsideRange(station, min, max, range) {
 return station.readings
  .filter(r => r.temp < min || r.temp > max);
}
```

JavaScript null

```
alerts = readingsOutsideRange(
  station,
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling,
  null
);
```

```
const range = new NumberRange(
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling
);
alerts = readingsOutsideRange(
  station,
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling,
  range
);
```

“ ”

```
function readingsOutsideRange(station, min, max, range) {
 return station.readings
  .filter(r => r.temp < min || r.temp > range.max);
}
```

```
const range = new NumberRange(
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling
);
alerts = readingsOutsideRange(
  station,
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling,
  range
);
```

```
function readingsOutsideRange(station, min, range) {
 return station.readings
  .filter(r => r.temp < range.min || r.temp > range.max);
}
```

```
const range = new NumberRange(
  operatingPlan.temperatureFloor,
  operatingPlan.temperatureCeiling
);
alerts = readingsOutsideRange(station, operatingPlan.temperatureFloor, range);
```

“ ”

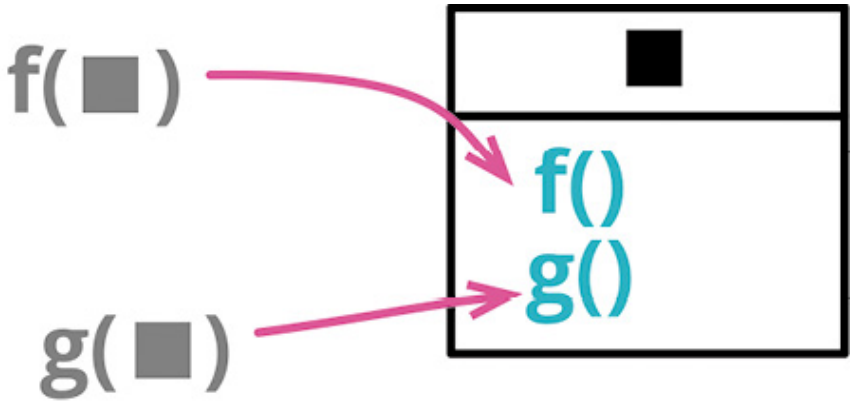
```
function readingsOutsideRange(station, range) {
 return station.readings
   .filter(r => !range.contains(r.temp));
}
```

**class NumberRange...**

```
  contains(arg) {return (arg >= this.min &amp;&amp; arg <= this.max);}
```

“ ”[mf-range] “ ” “ / ”
“ ” “ ” temperatureFloor temperatureCeiling temperatureRange
“ ” “ ”

## 6.9 Combine Functions into Class




```
function base(aReading) {...}
function taxableCharge(aReading) {...}
function calculateBaseCharge(aReading) {...}
```

```
class Reading {
  base() {...}
  taxableCharge() {...}
  calculateBaseCharge() {...}
}
```

149

“ ” Function As Object [mf-fao]

- 162

> **Tip**
>
> 140

- 198

> **Tip**

- 106

reading

```
reading = { customer: "ivan", quantity: 10, month: 5, year: 2017 };
```

“ ” base charge

### 1...

```
const aReading = acquireReading();
const baseCharge = baseRate(aReading.month, aReading.year) * aReading.quantity
```

### 2...

```
const aReading = acquireReading();
const base = baseRate(aReading.month, aReading.year) * aReading.quantity;
const taxableCharge = Math.max(0, base - taxThreshold(aReading.year));
```

106

### 3...

```
const aReading = acquireReading();
const basicChargeAmount = calculateBaseCharge(aReading);

function calculateBaseCharge(aReading) {
  return baseRate(aReading.month, aReading.year) * aReading.quantity;
}
```

162

```
class Reading {
  constructor(data) {
    this._customer = data.customer;
    this._quantity = data.quantity;
    this._month = data.month;
    this._year = data.year;
  }
  get customer() {
    return this._customer;
  }
  get quantity() {
    return this._quantity;
  }
  get month() {
    return this._month;
  }
  get year() {
    return this._year;
  }
}
```

calculateBaseCharge Reading Reading Reading

### 3...

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const basicChargeAmount = calculateBaseCharge(aReading);
```

198 calculateBaseCharge

## class Reading...

```
  get calculateBaseCharge() {
  return baseRate(this.month, this.year) * this.quantity;
}
```

### 3...

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const basicChargeAmount = aReading.calculateBaseCharge;
```

124

```
  get baseCharge() {
  return baseRate(this.month, this.year) * this.quantity;
}
```

### 3...

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const basicChargeAmount = aReading.baseCharge;
```

Reading baseCharge “ ” Uniform Access Principle [mf-ua]

1

### 1...

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const baseCharge = aReading.baseCharge;
```

123 baseCharge “ ” baseCharge

### 2...

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const taxableCharge = Math.max(
  0,
  aReading.baseCharge - taxThreshold(aReading.year)
);
```

106 taxable charge

```
function taxableChargeFn(aReading) {
  return Math.max(0, aReading.baseCharge - taxThreshold(aReading.year));
}
```

**3...**

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const taxableCharge = taxableChargeFn(aReading);
```

198 Reading

**class Reading...**

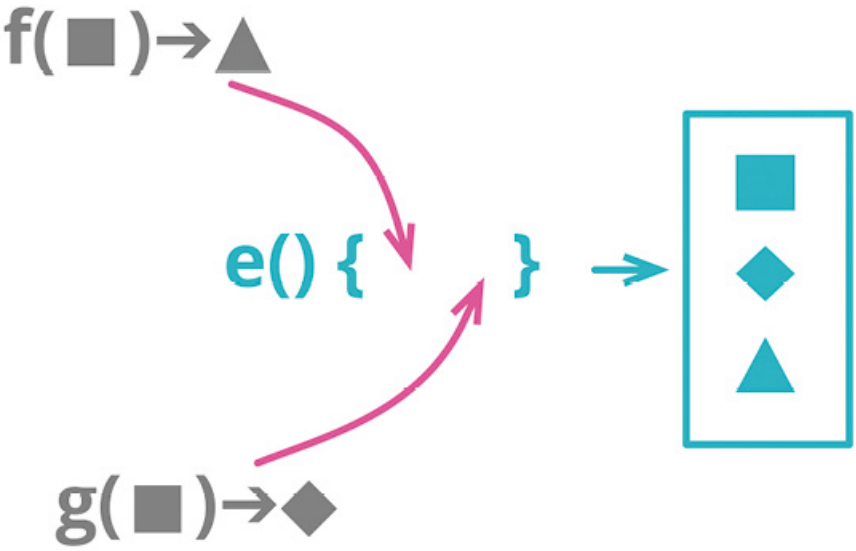
```
get taxableCharge() {
  return Math.max(0, this.baseCharge - taxThreshold(this.year));
}
```

**3...**

```
const rawReading = acquireReading();
const aReading = new Reading(rawReading);
const taxableCharge = aReading.taxableCharge;
```

JavaScript

# 6.10 Combine Functions into Transform




```
function base(aReading) {...}
function taxableCharge(aReading) {...}
```

```
function enrichReading(argReading) {
  const aReading = _.cloneDeep(argReading);
  aReading.baseCharge = base(aReading);
  aReading.taxableCharge = taxableCharge(aReading);
  return aReading;
}
```

“ ”

transform

144

106

- 

> **Tip**
>
> deep copy

- 

> **Tip**
>
> 106

- 
- 

reading

```
reading = { customer: "ivan", quantity: 10, month: 5, year: 2017 };
```

### 1...

```
const aReading = acquireReading();
const baseCharge = baseRate(aReading.month, aReading.year) * aReading.quantity
```

—

### 2...

```
const aReading = acquireReading();
const base = baseRate(aReading.month, aReading.year) * aReading.quantity;
const taxableCharge = Math.max(0, base - taxThreshold(aReading.year));
```

106

### 3...

```
const aReading = acquireReading();
const basicChargeAmount = calculateBaseCharge(aReading);

function calculateBaseCharge(aReading) {
  return baseRate(aReading.month, aReading.year) * aReading.quantity;
}
```

“ ” “ ”

```
function enrichReading(original) {
  const result = _.cloneDeep(original);
  return result;
}
```

Lodash cloneDeep

“enrich”

“transform”

enrichReading “ ”

### 3...

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const basicChargeAmount = calculateBaseCharge(aReading);
```

198 calculateBaseCharge

```
function enrichReading(original) {
  const result = _.cloneDeep(original);
  result.baseCharge = calculateBaseCharge(result);
  return result;
}
```

aReading accumulating variable

enrichReading

### 3...

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const basicChargeAmount = aReading.baseCharge;
```

calculateBaseCharge enrichReading “ ”

enrichReading

```
it("check reading unchanged", function () {
  const baseReading = { customer: "ivan", quantity: 15, month: 5, year: 2017 };
  const oracle = _.cloneDeep(baseReading);
  enrichReading(baseReading);
  assert.deepEqual(baseReading, oracle);
});
```

1

### 1...

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const baseCharge = aReading.baseCharge;
```

123 baseCharge

“ ”

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const base = baseRate(aReading.month, aReading.year) * aReading.quantity;
const taxableCharge = Math.max(0, base - taxThreshold(aReading.year));
```

106

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const base = aReading.baseCharge;
const taxableCharge = Math.max(0, base - taxThreshold(aReading.year));
```

123 base

```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const taxableCharge = Math.max(
  0,
  aReading.baseCharge - taxThreshold(aReading.year)
);
```

```
function enrichReading(original) {
  const result = _.cloneDeep(original);
  result.baseCharge = calculateBaseCharge(result);
  result.taxableCharge = Math.max(
    0,
    result.baseCharge - taxThreshold(result.year)
  );
  return result;
}
```

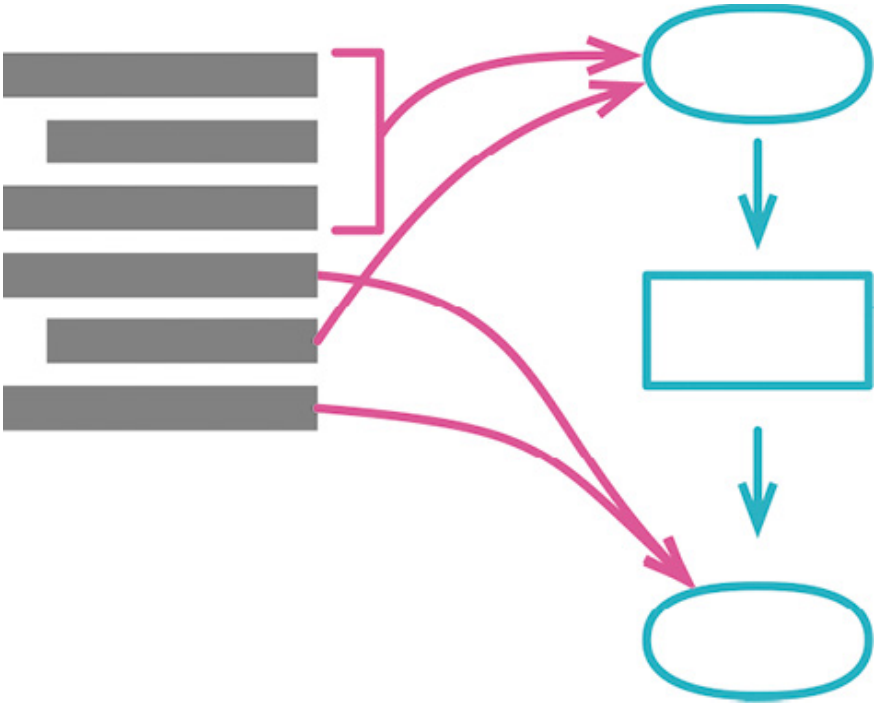
```
const rawReading = acquireReading();
const aReading = enrichReading(rawReading);
const taxableCharge = aReading.taxableCharge;
```

123 taxableCharge

quantity JavaScript

144

## 6.11 Split Phase



```
const orderData = orderString.split(/\s+/);
const productPrice = priceList[orderData[0].split("-")[1]];
const orderPrice = parseInt(orderData[1]) * productPrice;
```

```
const orderRecord = parseOrder(order);
const orderPrice = price(orderRecord, priceList);

function parseOrder(aString) {
  const values = aString.split(/\s+/);
  return {
    productID: values[0].split("-")[1],
    quantity: parseInt(values[1]),
  };
}
function price(order, priceList) {
  return order.quantity * priceList[order.productID];
}
```

token

- 
- 
- 
- 
- “ ”

> **Tip**
>
> 217 “ ”

- 106

> **Tip**
>
> transform

“ ”

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const shippingPerCase = (basePrice > shippingMethod.discountThreshold)
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = quantity * shippingPerCase;
 const price = basePrice - discount + shippingCost;
 return price;
}
```

product shipping

106

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const price = applyShipping(basePrice, shippingMethod, quantity, discount);
 return price;
}
function applyShipping(basePrice, shippingMethod, quantity, discount) {
 const shippingPerCase = (basePrice > shippingMethod.discountThreshold)
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = quantity * shippingPerCase;
 const price = basePrice - discount + shippingCost;
 return price;
}
```

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const priceData = {};
 const price = applyShipping(priceData, basePrice, shippingMethod, quantity, di
 return price;
}

function applyShipping(priceData, basePrice, shippingMethod, quantity, discount
 const shippingPerCase = (basePrice > shippingMethod.discountThreshold)
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;

 const shippingCost = quantity * shippingPerCase;
 const price = basePrice - discount + shippingCost;
 return price;
}
```

applyShipping basePrice

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const priceData = {basePrice: basePrice};
 const price = applyShipping(priceData, basePrice, shippingMethod, quantity, di
 return price;
}
function applyShipping(priceData, basePrice, shippingMethod, quantity, discount
 const shippingPerCase = (priceData.basePrice > shippingMethod.discountThreshol
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = quantity * shippingPerCase;
 const price = priceData.basePrice - discount + shippingCost;
 return price;
}
```

shippingMethod

quantity

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const priceData = {basePrice: basePrice, quantity: quantity};
 const price = applyShipping(priceData, shippingMethod, quantity, discount);
 return price;
}
function applyShipping(priceData, shippingMethod, quantity, discount) {
 const shippingPerCase = (priceData.basePrice > shippingMethod.discountThreshol
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = priceData.quantity * shippingPerCase;
 const price = priceData.basePrice - discount + shippingCost;
 return price;
}
```

discount

```
function priceOrder(product, quantity, shippingMethod) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 const priceData = {basePrice: basePrice, quantity: quantity, discount:discount
 const price = applyShipping(priceData, shippingMethod, discount);
 return price;
}
function applyShipping(priceData, shippingMethod, discount) {
 const shippingPerCase = (priceData.basePrice > shippingMethod.discountThreshol
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = priceData.quantity * shippingPerCase;
 const price = priceData.basePrice - priceData.discount + shippingCost;
 return price;
}
```

```
function priceOrder(product, quantity, shippingMethod) {
 const priceData = calculatePricingData(product, quantity);
 const price = applyShipping(priceData, shippingMethod);
 return price;
}
function calculatePricingData(product, quantity) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 return {basePrice: basePrice, quantity: quantity, discount:discount};
}
function applyShipping(priceData, shippingMethod) {
 const shippingPerCase = (priceData.basePrice > shippingMethod.discountThreshol
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = priceData.quantity * shippingPerCase;
 const price = priceData.basePrice - priceData.discount + shippingCost;
 return price;
}
```

const

```
function priceOrder(product, quantity, shippingMethod) {
 const priceData = calculatePricingData(product, quantity);
 return applyShipping(priceData, shippingMethod);
}


function calculatePricingData(product, quantity) {
 const basePrice = product.basePrice * quantity;
 const discount = Math.max(quantity - product.discountThreshold, 0)
     * product.basePrice * product.discountRate;
 return {basePrice: basePrice, quantity: quantity, discount:discount};
}
function applyShipping(priceData, shippingMethod) {
 const shippingPerCase = (priceData.basePrice > shippingMethod.discountThresho
     ? shippingMethod.discountedFee : shippingMethod.feePerCase;
 const shippingCost = priceData.quantity * shippingPerCase;
 return priceData.basePrice - priceData.discount + shippingCost;
}
```

# 7

[Parnas] 162 170
174 ——
178

6 144 / 182 186

189 —— 192

106

195

## 7.1 Encapsulate Record

Replace Record with Data Class

```
organization = { name: "Acme Gooseberries", country: "GB" };

class Organization {
  constructor(data) {
    this._name = data.name;
    this._country = data.country;
  }
  get name() {
    return this._name;
  }
  set name(arg) {
    this._name = arg;
  }
  get country() {
    return this._country;
  }
  set country(arg) {
    this._country = arg;
  }
}
```

“ ” “ ” {start: 1, end: 5} {start: 1, length: 5}
{end: 5, length: 5} 3 start end
length

3

3

hash map

hashmap dictionary associative array

/ /

“ ” ____

list JSON XML

132

162 170

```
const organization = { name: "Acme Gooseberries", country: "GB" };
```

JavaScript

```
result += `<h1>${organization.name}</h1>`;
organization.name = newName;
```

132

```
function getRawDataOfOrganization() {
  return organization;
}
```

•••

```
result += `<h1>${getRawDataOfOrganization().name}</h1>`;
```

•••

```
getRawDataOfOrganization().name = newName;
```

132

## class Organization...

```
class Organization {
  constructor(data) {
    this._data = data;
  }
}
```

```
const organization = new Organization({
  name: "Acme Gooseberries",
  country: "GB",
});

function getRawDataOfOrganization() {
  return organization._data;
}
function getOrganization() {
  return organization;
}
```

## class Organization...

```
  set name(aString) {this._data.name = aString;}
```

...

```
getOrganization().name = newName;
```

## class Organization...

```
get name() {return this._data.name;}
```

...

```
result += `<h1>${getOrganization().name}</h1>`;
```

```
function getRawDataOfOrganization() {
  return organization._data;
}
function getOrganization() {
  return organization;
}
```

_data

```
class Organization {
  constructor(data) {
    this._name = data.name;
    this._country = data.country;
  }
  get name() {
    return this._name;
  }
  set name(aString) {
    this._name = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}
```

_data

JSON

ID

```
"1920": {
name: "martin",
id: "1920",
usages: {
"2016": {
 "1": 50,
 "2": 55,
 // remaining months of the year
},
"2015": {
 "1": 70,
 "2": 63,
 // remaining months of the year
}
}
},
"38673": {
name: "neal",
id: "38673",
// more customers in a similar form
```

•••

```
customerData[customerID].usages[year][month] = amount;
```

•••

```
function compareUsage(customerID, laterYear, month) {
  const later = customerData[customerID].usages[laterYear][month];
  const earlier = customerData[customerID].usages[laterYear - 1][month];
  return { laterAmount: later, change: later - earlier };
}
```

132

```
function getRawDataOfCustomers() {
  return customerData;
}
function setRawDataOfCustomers(arg) {
  customerData = arg;
}
```

•••

```
getRawDataOfCustomers()[customerID].usages[year][month] = amount;
```

•••

```
function compareUsage(customerID, laterYear, month) {
  const later = getRawDataOfCustomers()[customerID].usages[laterYear][month];
  const earlier = getRawDataOfCustomers()[customerID].usages[laterYear - 1][
    month
  ];
  return { laterAmount: later, change: later - earlier };
}
```

```
class CustomerData {
  constructor(data) {
    this._data = data;
  }
}
```

•••

```
function getCustomerData() {
  return customerData;
}
function getRawDataOfCustomers() {
  return customerData._data;
}
function setRawDataOfCustomers(arg) {
  customerData = new CustomerData(arg);
}
```

getRawDataOfCustomers

•••

```
getRawDataOfCustomers()[customerID].usages[year][month] = amount;
```

“ ”

106

•••

```
setUsage(customerID, year, month, amount);
```

...

```
function setUsage(customerID, year, month, amount) {
  getRawDataOfCustomers()[customerID].usages[year][month] = amount;
}
```

198

...

```
getCustomerData().setUsage(customerID, year, month, amount);
```

## class CustomerData...

```
  setUsage(customerID, year, month, amount) {
  this._data[customerID].usages[year][month] = amount;
}
```

getRawDataOfCustomers

...

```
function getCustomerData() {
  return customerData;
}
function getRawDataOfCustomers() {
  return customerData.rawData;
}
function setRawDataOfCustomers(arg) {
  customerData = new CustomerData(arg);
}
```

## class CustomerData...

```
get rawData() {
  return _.cloneDeep(this._data);
}
```

lodash

JavaScript

CustomerData

## class CustomerData...

```
  usage(customerID, year, month) {
  return this._data[customerID].usages[year][month];
}
```

...

```
function compareUsage(customerID, laterYear, month) {
  const later = getCustomerData().usage(customerID, laterYear, month);
  const earlier = getCustomerData().usage(customerID, laterYear - 1, month);
  return { laterAmount: later, change: later - earlier };
}
```

customerData API
[mf-lh]

rawData

## class CustomerData...

```
get rawData() {
  return _.cloneDeep(this._data);
}
```

...

```
function compareUsage(customerID, laterYear, month) {
  const later = getCustomerData().rawData[customerID].usages[laterYear][month];
  const earlier = getCustomerData().rawData[customerID].usages[laterYear - 1][
    month
  ];
  return { laterAmount: later, change: later - earlier };
}
```

——

customer usage 170
usage 252

“Refactoring
Code to Load a Document”[mf-ref-doc]

# 7.2 Encapsulate Collection

```
class Person {
  get courses() {return this._courses;}
  set courses(aList) {this._courses = aList;}


class Person {
  get courses() {return this._courses.slice();}
  addCourse(aCourse) { ... }
  removeCourse(aCourse) { ... }
```

——— ———

——— “ ” “ ”

bug

aCustomer.orders.size

aCustomer.numberOfOrders Collection

Pipeline [mf-cp]

Java ——Java

——— 2.8

132

“ ” “ ”

331

/

Person Course “ ”

## class Person...

```
  constructor (name) {
  this._name = name;
  this._courses = [];
}
get name() {return this._name;}
get courses() {return this._courses;}
set courses(aList) {this._courses = aList;}
```

## class Course...

```
  constructor(name, isAdvanced) {
  this._name = name;
  this._isAdvanced = isAdvanced;
}
get name() {return this._name;}
get isAdvanced() {return this._isAdvanced;}
```

```
  numAdvancedCourses = aPerson.courses
  .f ilter(c =&amp;gt; c.isAdvanced)
  .length
;
```

•••

```
  const basicCourseNames = readBasicCourseNames(filename);
aPerson.courses = basicCourseNames.map(name => new Course(name, false));
```

•••

```
for (const name of readBasicCourseNames(filename)) {
  aPerson.courses.push(new Course(name, false));
}
```

Person

“ ” “ ”

### class Person...

```
  addCourse(aCourse) {
  this._courses.push(aCourse);
}
removeCourse(aCourse, fnIfAbsent = () => {throw new RangeError();}) {
  const index = this._courses.indexOf(aCourse);
  if (index === -1) fnIfAbsent();
  else this._courses.splice(index, 1);
}
```

...

```
for (const name of readBasicCourseNames(filename)) {
  aPerson.addCourse(new Course(name, false));
}
```

setCourse 331 API

### class Person...

```
  set courses(aList) {this._courses = aList.slice();}
```

### class Person...

```
get courses() {return this._courses.slice();}
```

JavaScript

sort()

## 7.3 Replace Primitive with Object

Replace Data Value with Object

Replace Type Code with Class

```
orders.filter(o => "high" === o.priority
              || "rush" === o.priority);


orders.filter(o => o.priority.higherThan(new Priority("normal")))
```

“ ”

“ ”“ ”

——

132

124

252 256

Order priority

## class Order...

```
constructor(data) {
this.priority = data.priority;
// more initialization
```

•••

```
highPriorityCount = orders.filter(o => "high" === o.priority
                                  || "rush" === o.priority)
                          .length;
```

## class Order...

```
get priority()        {return this._priority;}
set priority(aString) {this._priority = aString;}
```

value class

```
class Priority {
  constructor(value) {
    this._value = value;
  }
  toString() {
    return this._value;
  }
}
```

toString value API “ ”

## class Order...

```
get priority()        {return this._priority.toString();}
set priority(aString) {this._priority = new Priority(aString);}
```

Priority Order Priority
124

## class Order...

```
get priorityString() {return this._priority.toString();}
set priority(aString) {this._priority = new Priority(aString);}
```

•••

```
highPriorityCount = orders.filter(o => "high" === o.priorityString
                                    || "rush" === o.priorityString)
                          .length;
```

Priority

Priority

## class Order...

```
get priority()        {return this._priority;}
get priorityString()  {return this._priority.toString();}
set priority(aString) {this._priority = new Priority(aString);}
```

...

```
highPriorityCount = orders.filter(o => "high" === o.priority.toString()
                                    || "rush" === o.priority.toString())
                          .length;
```

Priority Order Priority Priority

## class Priority...

```
constructor(value) {
  if (value instanceof Priority) return value;
  this._value = value;
}
```

Priority ——

## class Priority...

```
constructor(value) {
  if (value instanceof Priority) return value;
  if (Priority.legalValues().includes(value))
    this._value = value;
  else
    throw new Error(`<${value}> is invalid for Priority`);
}
toString() {return this._value;}
get _index() {return Priority.legalValues().findIndex(s => s === this._value);}
static legalValues() {return ['low', 'normal', 'high', 'rush'];}

equals(other) {return this._index === other._index;}
higherThan(other) {return this._index > other._index;}
lowerThan(other) {return this._index < other._index;}
```

value object equals

...

```
  highPriorityCount = orders.filter(o => o.priority.higherThan(new Priority("no
                            .length;
```

## 7.4 Replace Temp with Query

```
  const basePrice = this._quantity * this._itemPrice;
if (basePrice > 1000)
  return basePrice * 0.95;
else
  return basePrice * 0.98;


  get basePrice() {this._quantity * this._itemPrice;}

...

if (this.basePrice > 1000)
  return this.basePrice * 0.95;
else
  return this.basePrice * 0.98;
```

178

---

oldAddress

306

123

## class Order...

```
    constructor(quantity, item) {
    this._quantity = quantity;
    this._item = item;
  }

get price() {
    var basePrice = this._quantity * this._item.price;
    var discountFactor = 0.98;
    if (basePrice > 1000) discountFactor -= 0.03;
    return basePrice * discountFactor;
  }
}
```

basePrice discountFactor

basePrice const ——

## class Order...

```
   constructor(quantity, item) {
  this._quantity = quantity;
  this._item = item;
 }

 get price() {
  const basePrice = this._quantity * this._item.price;
  var discountFactor = 0.98;
  if (basePrice > 1000) discountFactor -= 0.03;
  return basePrice * discountFactor;
 }
}
```

## class Order...

```
get price() {
 const basePrice = this.basePrice;
 var discountFactor = 0.98;
 if (basePrice > 1000) discountFactor -= 0.03;
 return basePrice * discountFactor;
}

 get basePrice() {
  return this._quantity * this._item.price;
 }
```

123

### class Order...

```
get price() {
 const basePrice = this.basePrice;
 var discountFactor = 0.98;
 if (this.basePrice > 1000) discountFactor -= 0.03;
 return this.basePrice * discountFactor;
}
```

discountFactor 106

### class Order...

```
get price() {
 const discountFactor = this.discountFactor;
 return this.basePrice * discountFactor;
}

 get discountFactor() {
  var discountFactor = 0.98;
  if (this.basePrice > 1000) discountFactor -= 0.03;
  return discountFactor;
}
```

discountFactor const

```
get price() {
  return this.basePrice * this.discountFactor;
}
```

## 7.5 Extract Class

186

```
  class Person {
 get officeAreaCode() {return this._officeAreaCode;}
 get officeNumber()   {return this._officeNumber;}


  class Person {
 get officeAreaCode() {return this._telephoneNumber.areaCode;}
 get officeNumber()   {return this._telephoneNumber.number;}
}
class TelephoneNumber {
 get areaCode() {return this._areaCode;}
 get number()   {return this._number;}
}
```

“ ”

207

198 “ ” “ ”

252

Person

## class Person...

```
get name()   {return this._name;}
set name(arg) {this._name = arg;}
get telephoneNumber() {return `(${this.officeAreaCode}) ${this.officeNumber}`;}
get officeAreaCode()   {return this._officeAreaCode;}
set officeAreaCode(arg) {this._officeAreaCode = arg;}
get officeNumber() {return this._officeNumber;}
set officeNumber(arg) {this._officeNumber = arg;}
```

TelephoneNumber “ ”

```
class TelephoneNumber {}
```

Person TelephoneNumber

## class Person...

```
constructor() {
  this._telephoneNumber = new TelephoneNumber();
}
```

## class TelephoneNumber...

```
get officeAreaCode()    {return this._officeAreaCode;}
set officeAreaCode(arg) {this._officeAreaCode = arg;}
```

207

## class Person...

```
get officeAreaCode()    {return this._telephoneNumber.officeAreaCode;}
set officeAreaCode(arg) {this._telephoneNumber.officeAreaCode = arg;}
```

## class TelephoneNumber...

```
get officeNumber() {return this._officeNumber;}
set officeNumber(arg) {this._officeNumber = arg;}
```

## class Person...

```
get officeNumber() {return this._telephoneNumber.officeNumber;}
set officeNumber(arg) {this._telephoneNumber.officeNumber = arg;}
```

### class TelephoneNumber...

```
get telephoneNumber() {return `(${this.officeAreaCode}) ${this.officeNumber}`;}
```

### class Person...

```
get telephoneNumber() {return this._telephoneNumber.telephoneNumber;}
```

“ ” “ ” office

### class TelephoneNumber...

```
get areaCode()    {return this._areaCode;}
set areaCode(arg) {this._areaCode = arg;}

get number()    {return this._number;}
set number(arg) {this._number = arg;}
```

### class Person...

```
get officeAreaCode()    {return this._telephoneNumber.areaCode;}
set officeAreaCode(arg) {this._telephoneNumber.areaCode = arg;}
get officeNumber()    {return this._telephoneNumber.number;}
set officeNumber(arg) {this._telephoneNumber.number = arg;}
```

TelephoneNumber telephone number 124

### class TelephoneNumber...

```
  toString() {return `(${this.areaCode}) ${this.number}`;}
```

### class Person...

```
get telephoneNumber() {return this._telephoneNumber.toString();}
```

“ ” TelephoneNumber Person office
TelephoneNumber Value Object [mf-vo] 252

## 7.6 Inline Class

182

```
  class Person {
 get officeAreaCode() {return this._telephoneNumber.areaCode;}
 get officeNumber()  {return this._telephoneNumber.number;}
}
class TelephoneNumber {
 get areaCode() {return this._areaCode;}
 get number() {return this._number;}
}


  class Person {
 get officeAreaCode() {return this._officeAreaCode;}
 get officeNumber()  {return this._officeNumber;}
```

182 “ ”
“ ”

182

public

public

“ ”

shipment tracking information

```
class TrackingInformation {
  get shippingCompany() {
    return this._shippingCompany;
  }
  set shippingCompany(arg) {
    this._shippingCompany = arg;
  }
  get trackingNumber() {
    return this._trackingNumber;
  }
  set trackingNumber(arg) {
    this._trackingNumber = arg;
  }
  get display() {
    return `${this.shippingCompany}: ${this.trackingNumber}`;
  }
}
```

Shipment

## class Shipment...

```
get trackingInfo() {
 return this._trackingInformation.display;
}
get trackingInformation() {return this._trackingInformation;}
set trackingInformation(aTrackingInformation) {
 this._trackingInformation = aTrackingInformation;
}
```

TrackingInformation Shipment

TrackingInformation

...

```
aShipment.trackingInformation.shippingCompany = request.vendor;
```

Shipment 198 Shipment

## class Shipment...

```
    set shippingCompany(arg) {this._trackingInformation.shippingCompany = arg;}
```

...

```
aShipment.trackingInformation.shippingCompany = request.vendor;
```

TrackingInformation Shipment

display 115

### class Shipment...

```
get trackingInfo() {
  return `${this.shippingCompany}: ${this.trackingNumber}`;
}
```

“ ” shipping company

```
get shippingCompany() {return this._trackingInformation._shippingCompany;}
set shippingCompany(arg) {this._trackingInformation._shippingCompany = arg;}
```

207 Shipment shippingCompany

TrackingInformation

### class Shipment...

```
get trackingInfo() {
 return `${this.shippingCompany}: ${this.trackingNumber}`;
}
get shippingCompany()    {return this._shippingCompany;}
set shippingCompany(arg) {this._shippingCompany = arg;}
get trackingNumber()    {return this._trackingNumber;}
set trackingNumber(arg) {this._trackingNumber = arg;}
```

## 7.7 Hide Delegate

192

```
  manager = aPerson.department.manager;


  manager = aPerson.manager;

class Person {
  get manager() {return this.department.manager;}
```

“ ” “ ” ____

Delegate

“ ” Person “ ” Department

## class Person...

```
  constructor(name) {
 this._name = name;
}
get name() {return this._name;}
get department()    {return this._department;}
set department(arg) {this._department = arg;}
```

## class Department...

```
get chargeCode() {return this._chargeCode;}
set chargeCode(arg) {this._chargeCode = arg;}
get manager() {return this._manager;}
set manager(arg) {this._manager = arg;}
```

Department

...

```
manager = aPerson.department.manager;
```

Department Department “ ” Department Person

## class Person...

```
get manager() {return this._department.manager;}
```

Person

•••

```
manager = aPerson.department.manager;
```

Department Person Person department

## 7.8 Remove Middle Man

189

```
  manager = aPerson.manager;

class Person {
 get manager() {return this.department.manager;}


  manager = aPerson.department.manager;
```

189 “ ” “ ”

81 “

”

189 “ ”

6 ——

132 115

Person “ ”

•••

```
manager = aPerson.manager;
```

### class Person...

```
get manager() {return this._department.manager;}
```

### class Department...

```
get manager() {return this._manager;}
```

Department Person Person

### class Person...

```
get department() {return this._department;}
```

...

```
manager = aPerson.department.manager;
```

Person manager Person

——

department 132 manager department

### class Person...

```
get manager() {return this.department.manager;}
```

JavaScript department

manager 115

## 7.9 Substitute Algorithm

```
  function foundPerson(people) {
 for(let i = 0; i < people.length; i++) {
  if (people[i] === "Don") {
   return "Don";
  }
  if (people[i] === "John") {
   return "John";
  }
  if (people[i] === "Kent") {
   return "Kent";
  }
 }
 return "";
}
```

```
  function foundPerson(people) {
 const candidates = ["Don", "John", "Kent"];
 return people.find(p => candidates.includes(p)) || '';
}
```

“ ”

/

- 
- 
- 
- 
- 

# 8

198

207

213 217 223

222

227 231

237

## 8.1 Move Function

Move Method

```
class Account {
get overdraftCharge() {...}


class AccountType {
  get overdraftCharge() {...}
```

144 182 “ ”

source context

115

GPS track record total distance

```
function trackSummary(points) {
 const totalTime = calculateTime();
 const totalDistance = calculateDistance();
 const pace = totalTime / 60 / totalDistance ;
 return {
  time: totalTime,
  distance: totalDistance,
  pace: pace
 };

 function calculateDistance() {
  let result = 0;
  for (let i = 1; i < points.length; i++) {
   result += distance(points[i-1],  points[i]);
  }
  return result;
 }

 function distance(p1,p2) { ... }
 function radians(degrees) { ... }
 function calculateTime() { ... }

}
```

calculateDistance summary

```
 function trackSummary(points) {
const totalTime = calculateTime();
const totalDistance = calculateDistance();
const pace = totalTime / 60 / totalDistance ;
return {
 time: totalTime,
 distance: totalDistance,
 pace: pace
};

function calculateDistance() {
 let result  =  0;
 for (let i = 1; i < points.length; i++) {
  result += distance(points[i-1], points[i]);
 }
 return result;
}
...
function distance(p1,p2) { ... }
function radians(degrees) { ... }
function calculateTime() { ... }

}

function top_calculateDistance() {
 let result  =  0;
 for (let i = 1; i < points.length; i++) {
  result += distance(points[i-1],  points[i]);
 }
 return result;
}
```

“ ”

distance points points

```
 function top_calculateDistance(points) {
let result =0;
for (let i = 1; i < points.length; i++) {
 result += distance(points[i-1],  points[i]);
}
return result;
}
```

distance calculate Distance

## function trackSummary...

```
function distance(p1, p2) {
  const EARTH_RADIUS = 3959; // in miles
  const dLat = radians(p2.lat) - radians(p1.lat);
  const dLon = radians(p2.lon) - radians(p1.lon);
  const a =
    Math.pow(Math.sin(dLat / 2), 2) +
    Math.cos(radians(p2.lat)) *
      Math.cos(radians(p1.lat)) *
      Math.pow(Math.sin(dLon / 2), 2);
  const c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
  return EARTH_RADIUS * c;
}
function radians(degrees) {
  return (degrees * Math.PI) / 180;
}
```

distance radians radians
calculateDistance

```
 function trackSummary(points) {
const totalTime = calculateTime();
const totalDistance = calculateDistance();
const pace = totalTime / 60 / totalDistance ;
return {
 time: totalTime,
 distance: totalDistance,
 pace: pace
};

function calculateDistance() {
 let result = 0;
 for (let i = 1; i < points.length; i++) {
  result += distance(points[i-1], points[i]);
 }
 return result;

 function distance(p1,p2) { ... }
 function radians(degrees) { ... }

}
```

top_calculateDistance

```
 function top_calculateDistance(points) {
let result = 0;
for (let i = 1; i < points.length; i++) {
 result += distance(points[i-1],  points[i]);
}
return result;

function distance(p1,p2) { ... }
function radians(degrees) { ... }

}
```

distance radians

—— calculateDistance top_calculateDistance

```
 function trackSummary(points) {
const totalTime = calculateTime();
const totalDistance = calculateDistance();
const pace = totalTime / 60 / totalDistance ;
return {
 time: totalTime,
 distance: totalDistance,
 pace: pace
};

function calculateDistance() {
 return top_calculateDistance(points);
}
```

```
function trackSummary(points) {
  const totalTime = calculateTime();
  const totalDistance = top_calculateDistance(points);
  const pace = totalTime / 60 / totalDistance;
  return {
    time: totalTime,
    distance: totalDistance,
    pace: pace,
  };
}
```

totalDistance trackSummary

—— 123 124

```
  function trackSummary(points) {
 const totalTime = calculateTime();
 const pace = totalTime / 60 / totalDistance(points) ;
 return {
  time: totalTime,
  distance: totalDistance(points),
  pace: pace
 };
}
function totalDistance(points) {
 let result = 0;
 for (let i = 1; i < points.length; i++) {
  result += distance(points[i-1], points[i]);
 }
 return result;
}
```

totalDistanceCache distance

distance radians totalDistance 4

```
  function trackSummary(points) { ... }
function totalDistance(points) { ... }
function distance(p1,p2) { ... }
function radians(degrees) { ... }
```

distance radians totalDistance ES 2015 JavaScript

“ ” Account

## class Account...

```
  get bankCharge() {
 let result = 4.5;
 if (this._daysOverdrawn > 0) result += this.overdraftCharge;
 return result;
}

get overdraftCharge() {
 if (this.type.isPremium) {
  const baseCharge = 10;
  if (this.daysOverdrawn <= 7)
   return baseCharge;
  else
   return baseCharge + (this.daysOverdrawn - 7) * 0.85;
 }
 else
  return this.daysOverdrawn * 1.75;
}
```

account type “ ” overdraftCharge AccountType

overdraftCharge overdraftCharge
daysOverdrawn Account

overdraftCharge AccountType

## class AccountType...

```
  overdraftCharge(daysOverdrawn) {
 if (this.isPremium) {
  const baseCharge  =  10;
  if (daysOverdrawn <= 7)
   return baseCharge;
  else
   return baseCharge + (daysOverdrawn - 7) * 0.85;
 }
 else
  return daysOverdrawn * 1.75;
}
```

isPremium this daysOverdrawn
account account daysOverdrawn
account type account account

## class Account...

```
  get bankCharge() {
 let result = 4.5;
 if (this._daysOverdrawn > 0) result += this.overdraftCharge;
 return result;
}

get overdraftCharge() {
 return this.type.overdraftCharge(this.daysOverdrawn);
}
```

overdraftCharge

## class Account...

```
  get bankCharge() {
 let result = 4.5;
 if (this._daysOverdrawn > 0)
  result += this.type.overdraftCharge(this.daysOverdrawn);
 return result;
}
```

daysOverdrawn overdraftCharge account

## class Account...

```
  get bankCharge() {
 let result = 4.5;
 if (this._daysOverdrawn > 0) result += this.overdraftCharge;
 return result;
}

get overdraftCharge() {
 return this.type.overdraftCharge(this);
}
```

class AccountType…

```
 overdraftCharge(account) {
if (this.isPremium) {
 const baseCharge = 10;
 if (account.daysOverdrawn <= 7)
  return baseCharge;
 else
  return baseCharge + (account.daysOverdrawn - 7) * 0.85;
}
else
 return account.daysOverdrawn * 1.75;
}
```

## 8.2 Move Field

```
 class Customer {
 get plan() {return this._plan;}
 get discountRate() {return this._discountRate;}


 class Customer {
 get plan() {return this._plan;}
 get discountRate() {return this.plan.discountRate;}
```

—— domain-driven design

“ ”

“ ” record

“ ”

302

Customer “ ” CustomerContract “ ”

## class Customer...

```
  constructor(name, discountRate) {
 this._name = name;
 this._discountRate = discountRate;
 this._contract = new CustomerContract(dateToday());
}
get discountRate() {return this._discountRate;}
becomePreferred() {
 this._discountRate += 0.03;
 // other nice things
}
applyDiscount(amount) {
 return amount.subtract(amount.multiply(this._discountRate));
}
```

## class CustomerContract...

```
  constructor(startDate) {
  this._startDate = startDate;
}
```

discountRate Customer CustomerContract

132 _discountRate

## class Customer...

```
  constructor(name, discountRate) {
 this._name = name;
 this._setDiscountRate(discountRate);
 this._contract = new CustomerContract(dateToday());
}
get discountRate() {return this._discountRate;}
_setDiscountRate(aNumber) {this._discountRate = aNumber;}
becomePreferred() {
 this._setDiscountRate(this.discountRate + 0.03);
 // other nice things
}
applyDiscount(amount)  {
 return amount.subtract(amount.multiply(this.discountRate));
}
```

applyDiscount public

CustomerContract

## class CustomerContract...

```
  constructor(startDate, discountRate) {
 this._startDate = startDate;
 this._discountRate = discountRate;
}
get discountRate()  {return this._discountRate;}
set discountRate(arg) {this._discountRate = arg;}
```

customer CustomerContract “Cannot set property 'discountRate' of undefined” `_setDiscountRate` CustomerContract 223 `_setDiscountRate`

## class Customer...

```
  constructor(name, discountRate) {
  this._name = name;
  this._setDiscountRate(discountRate);
  this._contract = new CustomerContract(dateToday());
}
```

Customer _contract discountRate

## class Customer...

```
  get discountRate() {return this._contract.discountRate;}
_setDiscountRate(aNumber) {this._contract.discountRate = aNumber;}
```

JavaScript

immutable

162

“ ” Account “ ” _interestRate

## class Account...

```
constructor(number, type, interestRate) {
 this._number = number;
 this._type = type;
 this._interestRate = interestRate;
}
get interestRate() {return this._interestRate;}
```

## class AccountType...

```
constructor(nameString) {
  this._name = nameString;
}
```

AccountType

AccountType

## class AccountType...

```
constructor(nameString, interestRate) {
  this._name = nameString;
  this._interestRate = interestRate;
}
get interestRate() {return this._interestRate;}
```

Account

Account 302

## class Account...

```
constructor(number, type, interestRate) {
  this._number = number;
  this._type = type;
  assert(interestRate === this._type.interestRate);
  this._interestRate = interestRate;
}
get interestRate() {return this._interestRate;}
```

Account AccountType interestRate

### class Account...

```
constructor(number, type) {
  this._number = number;
  this._type = type;
}
get interestRate() {return this._type.interestRate;}
```

## 8.3 Move Statements into Function

217

```
result.push(`<p>title: ${person.photo.title}</p>`);
result.concat(photoData(person.photo));

function photoData(aPhoto) {
  return [
    `<p>location: ${aPhoto.location}</p>`,
    `<p>date: ${aPhoto.date.toDateString()}</p>`,
  ];
}

result.concat(photoData(person.photo));

function photoData(aPhoto) {
  return [
    `<p>title: ${aPhoto.title}</p>`,
    `<p>location: ${aPhoto.location}</p>`,
    `<p>date: ${aPhoto.date.toDateString()}</p>`,
  ];
}
```

“ ”

217

106

223

106

115

124

photo HTML

```
function renderPerson(outStream, person) {
  const result = [];
  result.push(`<p>${person.name}</p>`);
  result.push(renderPhoto(person.photo));
  result.push(`<p>title: ${person.photo.title}</p>`);
  result.push(emitPhotoData(person.photo));
  return result.join("\n");
}
function photoDiv(p) {
  return [
    "<div>",
    `<p>title:  ${p.title}</p>`,
    emitPhotoData(p),
    "</div>",
  ].join("\n");
}

function emitPhotoData(aPhoto) {
  const result = [];
  result.push(`<p>location: ${aPhoto.location}</p>`);
  result.push(`<p>date: ${aPhoto.date.toDateString()}</p>`);
  return result.join("\n");
}
```

emitPhotoData title
emitPhotoData emitPhotoData

106 emitPhotoData

```
function photoDiv(p) {
  return ["<div>", zznew(p), "</div>"].join("\n");
}

function zznew(p) {
  return [`<p>title: ${p.title}</p>`, emitPhotoData(p)].join("\n");
}
```

emitPhotoData

```
function renderPerson(outStream, person) {
  const result = [];
  result.push(`<p>${person.name}</p>`);
  result.push(renderPhoto(person.photo));
  result.push(zznew(person.photo));
  return result.join("\n");
}
```

emitPhotoData 115 emitPhotoData

```
function zznew(p) {
  return [
    `<p>title: ${p.title}</p>`,
    `<p>location: ${p.location}</p>`,
    `<p>date: ${p.date.toDateString()}</p>`,
  ].join("\n");
}
```

124

```
function renderPerson(outStream, person) {
  const result = [];
  result.push(`<p>${person.name}</p>`);
  result.push(renderPhoto(person.photo));
  result.push(emitPhotoData(person.photo));
  return result.join("\n");
}

function photoDiv(aPhoto) {
  return ["<div>", emitPhotoData(aPhoto), "</div>"].join("\n");
}

function emitPhotoData(aPhoto) {
  return [
    `<p>title: ${aPhoto.title}</p>`,
    `<p>location: ${aPhoto.location}</p>`,
    `<p>date: ${aPhoto.date.toDateString()}</p>`,
  ].join("\n");
}
```

## 8.4 Move Statements to Callers

213

```
emitPhotoData(outStream, person.photo);

function emitPhotoData(outStream, photo) {
  outStream.write(`<p>title: ${photo.title}</p>\n`);
  outStream.write(`<p>location: ${photo.location}</p>\n`);
}

emitPhotoData(outStream, person.photo);
outStream.write(`<p>location: ${person.photo.location}</p>\n`);

function emitPhotoData(outStream, photo) {
  outStream.write(`<p>title: ${photo.title}</p>\n`);
}
```

223

115

106

115

124

emitPhotoData

```
  function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 emitPhotoData(outStream, person.photo);
}

function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   emitPhotoData(outStream, p);
   outStream.write("</div>\n");
  });
}

function emitPhotoData(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
 outStream.write(`<p>location: ${photo.location}</p>\n`);
}
```

listRecentPhotos location renderPerson emitPhotoData

emitPhotoData

106 emitPhotoData

```
  function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 emitPhotoData(outStream, person.photo);
}

function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   emitPhotoData(outStream, p);
   outStream.write("</div>\n");
  });
}

function  emitPhotoData(outStream, photo) {
 zztmp(outStream,  photo);
 outStream.write(`<p>location: ${photo.location}</p>\n`);
}

function zztmp(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
}
```

emitPhotoData 115 renderPerson

```
  function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 zztmp(outStream,  person.photo);
 outStream.write(`<p>location: ${person.photo.location}</p>\n`);
}
function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   emitPhotoData(outStream, p);
   outStream.write("</div>\n");
  });
}

function emitPhotoData(outStream, photo) {
 zztmp(outStream, photo);
 outStream.write(`<p>location: ${photo.location}</p>\n`);
}

function zztmp(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
}
```

```
  function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 zztmp(outStream,  person.photo);
 outStream.write(`<p>location: ${person.photo.location}</p>\n`);
}

function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   zztmp(outStream, p);
   outStream.write(`<p>location: ${p.location}</p>\n`);
   outStream.write("</div>\n");
  });
}

function emitPhotoData(outStream, photo) {
 zztmp(outStream, photo);
 outStream.write(`<p>location: ${photo.location}</p>\n`);
}

function zztmp(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
}
```

emitPhotoData 115

```
function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 zztmp(outStream,  person.photo);
 outStream.write(`<p>location: ${person.photo.location}</p>\n`);
}

function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   zztmp(outStream, p);
   outStream.write(`<p>location: ${p.location}</p>\n`);
   outStream.write("</div>\n");
  });
}

function emitPhotoData(outStream, photo) {
 zztmp(outStream, photo);
 outStream.write(`<p>location: ${photo.location}</p>\n`);
}

function zztmp(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
}
```

zztmp emitPhotoData

```
function renderPerson(outStream, person) {
 outStream.write(`<p>${person.name}</p>\n`);
 renderPhoto(outStream, person.photo);
 emitPhotoData(outStream, person.photo);
 outStream.write(`<p>location: ${person.photo.location}</p>\n`);
}

function listRecentPhotos(outStream, photos) {
 photos
  .filter(p => p.date > recentDateCutoff())
  .forEach(p => {
   outStream.write("<div>\n");
   emitPhotoData(outStream, p);
   outStream.write(`<p>location: ${p.location}</p>\n`);
   outStream.write("</div>\n");
  });
}

function emitPhotoData(outStream, photo) {
 outStream.write(`<p>title: ${photo.title}</p>\n`);
 outStream.write(`<p>date: ${photo.date.toDateString()}</p>\n`);
}
```

## 8.5 Replace Inline Code with Function Call

```
let appliesToMass = false;
for (const s of states) {
  if (s === "MA") appliesToMass = true;
}

appliesToMass = states.includes("MA");
```

———

## 8.6 Slide Statements

Consolidate Duplicate Conditional Fragments

```
const pricingPlan = retrievePricingPlan();
const order = retreiveOrder();
let charge;
const chargePerUnit = pricingPlan.unit;

const pricingPlan = retrievePricingPlan();
const chargePerUnit = pricingPlan.unit;
const order = retreiveOrder();
let charge;
```

“ ”

106

106

106

```
1 const pricingPlan = retrievePricingPlan();
2 const order = retreiveOrder();
3 const baseCharge = pricingPlan.base;
4 let charge;
5 const chargePerUnit = pricingPlan.unit;
6 const units = order.units;
7 let discount;
8 charge = baseCharge + units * chargePerUnit;
9 let discountableUnits = Math.max(units - pricingPlan.discountThreshold, 0);
10 discount = discountableUnits * pricingPlan.discountFactor;
11 if (order.isRepeat) discount += 20;
12 charge = charge - discount;
13 chargeOrder(charge);
```

discount 7 let discount 10 discount = ... discount ——— 106

order 2 const order = ... 6 const units = ...

2 retrieveOrder() …… Command-Query Separation [mf-cqs]

11 if(order.isRepeat)... 12 discount 11 13 chargeOrder(charge) 13 charge 12 8 charge = baseCharge + ... 9 11

```
a = a + 10;
a = a + 5;
```

charge 240

——

5 10

```
let result;
if (availableResources.length === 0) {
  result = createResource();
  allocatedResources.push(result);
} else {
  result = availableResources.pop();
  allocatedResources.push(result);
}
return result;
```

if-else

```
let result;
if (availableResources.length === 0) {
  result = createResource();
} else {
  result = availableResources.pop();
}
allocatedResources.push(result);
return result;
```

“ ” Swap Statement [wake-swap]

## 8.7 Split Loop

```
let averageAge = 0;
let totalSalary = 0;
for (const p of people) {
  averageAge += p.age;
  totalSalary += p.salary;
}
averageAge = averageAge / people.length;

let totalSalary = 0;
for (const p of people) {
  totalSalary += p.salary;
}

let averageAge = 0;
for (const p of people) {
  averageAge += p.age;
}
averageAge = averageAge / people.length;
```

106

2.8

106

total salary youngest

```
let youngest = people[0] ? people[0].age : Infinity;
let totalSalary = 0;
for (const p of people) {
 if (p.age < youngest) youngest = p.age;
 totalSalary += p.salary;
}

return `youngestAge: ${youngest}, totalSalary: ${totalSalary}`;
```

```
let youngest = people[0] ? people[0].age : Infinity;
let totalSalary = 0;
for (const p of people) {
 if (p.age < youngest) youngest = p.age;
 totalSalary += p.salary;
}
for (const p of people) {
 if (p.age < youngest) youngest = p.age;
 totalSalary += p.salary;
}

return `youngestAge: ${youngest}, totalSalary: ${totalSalary}`;
```

```
let youngest = people[0] ? people[0].age : Infinity;
let totalSalary = 0;
for (const p of people) {
 if (p.age < youngest) youngest = p.age;
 totalSalary += p.salary;
}

for (const p of people) {
 if (p.age < youngest) youngest = p.age;
 totalSalary += p.salary;
}

return `youngestAge: ${youngest}, totalSalary: ${totalSalary}`;
```

———

223

```
let totalSalary = 0;
for (const p of people) {
 totalSalary += p.salary;
}

let youngest = people[0] ? people[0].age : Infinity;
for (const p of people) {
 if (p.age < youngest) youngest = p.age;
}

return `youngestAge: ${youngest}, totalSalary: ${totalSalary}`;
```

106

```
  return `youngestAge: ${youngestAge()}, totalSalary: ${totalSalary()}`;

function totalSalary() {
 let totalSalary = 0;
 for (const p of people) {
  totalSalary += p.salary;
 }
 return totalSalary;
}

function youngestAge() {
 let youngest = people[0] ? people[0].age : Infinity;
 for (const p of people) {
  if (p.age < youngest) youngest = p.age;
 }
 return youngest;
}
```

totalSalary 231 youngestAge 195

```
  return `youngestAge: ${youngestAge()}, totalSalary: ${totalSalary()}`;

function totalSalary() {
 return people.reduce((total,p) => total + p.salary, 0);
}
function youngestAge() {
 return Math.min(...people.map(p => p.age));
}
```

## 8.8 Replace Loop with Pipeline

```
const names = [];
for (const i of input) {
  if (i.job === "programmer")
    names.push(i.name);
}


const names = input
  .filter(i => i.job === "programmer")
  .map(i => i.name)
;
```

collection

pipeline [mf-cp] map

filter map filter

——

CSV office

```
office, country, telephone
Chicago, USA, +1 312 373 1000
Beijing, China, +86 4008 900 505
Bangalore, India, +91 80 4064 9570
Porto Alegre, Brazil, +55 51 3079 3550
Chennai, India, +91 44 660 44766

... (more data follows)
```

acquireData city telephone number

```
function acquireData(input) {
  const lines = input.split("\n");
  let firstLine = true;
  const result = [];
  for (const line of lines) {
    if (firstLine) {
      firstLine = false;
      continue;
    }
    if (line.trim() === "") continue;
    const record = line.split(",");
    if (record[1].trim() === "India") {
      result.push({ city: record[0].trim(), phone: record[2].trim() });
    }
  }
  return result;
}
```

```
function acquireData(input) {
  const lines = input.split("\n");
  let firstLine = true;
  const result = [];
  const loopItems = lines;
  for (const line of loopItems) {
    if (firstLine) {
      firstLine = false;
      continue;
    }
    if (line.trim() === "") continue;
    const record = line.split(",");
    if (record[1].trim() === "India") {
      result.push({ city: record[0].trim(), phone: record[2].trim() });
    }
  }
  return result;
}
```

CSV slice slice

```
function acquireData(input) {
  const lines = input.split("\n");
  let firstLine = true;
  const result = [];
  const loopItems = lines.slice(1);
  for (const line of loopItems) {
    if (firstLine) {
      firstLine = false;
      continue;
    }
    if (line.trim() === "") continue;
    const record = line.split(",");
    if (record[1].trim() === "India") {
      result.push({ city: record[0].trim(), phone: record[2].trim() });
    }
  }
  return result;
}
```

firstLine ——

filter

```
function acquireData(input) {
 const lines = input.split("\n");
 const result = [];
 const loopItems = lines
    .slice(1)
    .filter(line => line.trim() !== "")
    ;
 for (const line of loopItems) {
  if (line.trim() === "") continue;
  const  record = line.split(",");
  if (record[1].trim() === "India") {
   result.push({city: record[0].trim(), phone: record[2].trim()});
  }
 }
 return result;
}
```

map record “ ”

```
function acquireData(input) {
 const lines = input.split("\n");
 const result = [];
 const loopItems = lines
    .slice(1)
    .filter(line => line.trim() !== "")
    .map(line => line.split(","))
    ;
 for (const line of loopItems) {
  const record = line;.split(",");
  if (record[1].trim() === "India") {
   result.push({city: record[0].trim(), phone: record[2].trim()});
  }
 }
 return result;
}
```

filter

```
function acquireData(input) {
 const lines = input.split("\n");
 const result = [];
 const loopItems = lines
    .slice(1)
    .filter(line => line.trim() !== "")
    .map(line => line.split(","))
    .filter(record => record[1].trim() === "India")
    ;
 for (const line of loopItems) {
  const record = line;
  if (record[1].trim() === "India") {
   result.push({city: record[0].trim(), phone: record[2].trim()});
  }
 }
 return result;
}
```

map

```
 function acquireData(input) {
 const lines = input.split("\n");
 const result = [];
 const loopItems = lines
    .slice(1)
    .filter(line => line.trim() !== "")
    .map(line => line.split(","))
    .filter(record => record[1].trim() === "India")
    .map(record => ({city: record[0].trim(), phone: record[2].trim()}))
    ;
 for (const line of loopItems) {
  const record = line;
  result.push(line);
 }
 return result;
}
```

```
function acquireData(input) {
 const lines = input.split("\n");
 const result = lines
    .slice(1)
    .filter(line => line.trim() !== "")
    .map(line => line.split(","))
    .filter(record => record[1].trim() === "India")
    .map(record => ({city: record[0].trim(), phone: record[2].trim()}))
    ;
 for (const line of loopItems) {
  const record = line;
  result.push(line);
 }
 return result;
}
```

result

```
function acquireData(input) {
 const lines = input.split("\n");
 return lines
    .slice (1)
    .filter (line => line.trim() !== "")
    .map   (line => line.split(","))
    .filter (fields => fields[1].trim() === "India")
    .map   (fields => ({city: fields[0].trim(), phone: fields[2].trim()}))
    ;
}
```

lines

“Refactoring with Loops and Collection Pipelines”[mf-ref-pipe]

## 8.9 Remove Dead Code

```
if (false) {
  doSomethingThatUsedToMatter();
}
```

———

# 9

bug

240 137

248

252 256

## 9.1 Split Variable

Remove Assignments to Parameters

Split Temp

```
let temp = 2 * (height + width);
console.log(temp);
temp = height * width;
console.log(temp);

const perimeter = 2 * (height + width);
console.log(perimeter);
const area = height * width;
console.log(area);
```

“ ” “ ” loop variable for

let i=0; i<10; i++ i collecting variable “ ”

“i=i+ ”

```
function distanceTravelled (scenario, time) {
 let result;
 let acc = scenario.primaryForce / scenario.mass;
 let primaryTime = Math.min(time, scenario.delay);
 result = 0.5 * acc * primaryTime * primaryTime;
 let secondaryTime = time - scenario.delay;
 if (secondaryTime > 0) {
  let primaryVelocity = acc * scenario.delay;
  acc = (scenario.primaryForce + scenario.secondaryForce) / scenario.mass;
  result += primaryVelocity * secondaryTime + 0.5 * acc * secondaryTime * secor
 }
 return result;
}
```

acc acc

symbol

const acc acc

```
function distanceTravelled (scenario, time) {
 let result;
 const primaryAcceleration = scenario.primaryForce / scenario.mass;
 let primaryTime = Math.min(time, scenario.delay);
 result = 0.5 * primaryAcceleration * primaryTime * primaryTime;
 let secondaryTime = time - scenario.delay;
 if (secondaryTime > 0) {

  let primaryVelocity = primaryAcceleration * scenario.delay;
  let acc = (scenario.primaryForce + scenario.secondaryForce) / scenario.mass;
  result += primaryVelocity * secondaryTime + 0.5 * acc * secondaryTime * secor
 }
 return result;
}
```

acc const acc acc

acc acc

```
function distanceTravelled (scenario, time) {
 let result;
 const primaryAcceleration = scenario.primaryForce / scenario.mass;
 let primaryTime = Math.min(time, scenario.delay);
 result = 0.5 * primaryAcceleration * primaryTime * primaryTime;
 let secondaryTime = time - scenario.delay;
 if (secondaryTime > 0) {
  let primaryVelocity = primaryAcceleration * scenario.delay;
  const secondaryAcceleration = (scenario.primaryForce + scenario.secondaryForc
  result += primaryVelocity * secondaryTime +
   0.5 * secondaryAcceleration * secondaryTime * secondaryTime;
 }
 return result;
}
```

—— 1

```
function discount (inputValue, quantity) {
 if (inputValue > 50) inputValue = inputValue - 2;
 if (quantity > 100) inputValue = inputValue - 1;
 return inputValue;
}
```

inputValue JavaScript inputValue

inputValue

```
function discount (originalInputValue, quantity) {
 let inputValue = originalInputValue;
 if (inputValue > 50) inputValue = inputValue - 2;
 if (quantity > 100) inputValue = inputValue - 1;
 return inputValue;
}
```

137

```
function discount (inputValue, quantity) {
 let result = inputValue;
 if (inputValue > 50) result = result - 2;
 if (quantity > 100) result = result - 1;
 return result;
}
```

inputValue inputValue result “
” “ ”____

1 haggis “ ……”
Martin Fowler ——

## 9.2 Rename Field

```
class  Organization {
  get name() {...}
}


class  Organization {
  get title() {...}
}
```

Fred Brooks “
”

162

124

124

```
const organization = { name: "Acme Gooseberries", country: "GB" };
```

name title name 162

```
class Organization {
  constructor(data) {
    this._name = data.name;
    this._country = data.country;
  }
  get name() {
    return this._name;
  }
  set name(aString) {
    this._name = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}

const organization = new Organization({
  name: "Acme Gooseberries",
  country: "GB",
});
```

4 4

____ “ ”

## class Organization...

```
class Organization {
  constructor(data) {
    this._title = data.name;
    this._country = data.country;
  }
  get name() {
    return this._title;
  }
  set name(aString) {
    this._title = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}
```

title

## class Organization...

```
class Organization {
  constructor(data) {
    this._title = data.title !== undefined ? data.title : data.name;
    this._country = data.country;
  }
  get name() {
    return this._title;
  }
  set name(aString) {
    this._title = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}
```

name title

```
const organization = new Organization({
  title: "Acme Gooseberries",
  country: "GB",
});
```

name title

## class Organization...

```
class Organization {
  constructor(data) {
    this._title = data.title;
    this._country = data.country;
  }
  get name() {
    return this._title;
  }
  set name(aString) {
    this._title = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}
```

124

## class Organization...

```
class Organization {
  constructor(data) {
    this._title = data.title;
    this._country = data.country;
  }
  get title() {
    return this._title;
  }
  set title(aString) {
    this._title = aString;
  }
  get country() {
    return this._country;
  }
  set country(aCountryCode) {
    this._country = aCountryCode;
  }
}
```

———

# 9.3 Replace Derived Variable with Query

```
get discountedTotal() {return this._discountedTotal;}
set discount(aNumber) {
 const old = this._discount;
 this._discount = aNumber;
 this._discountedTotal += old - aNumber;
}


get discountedTotal() {return this._baseTotal - this._discount;}
set discount(aNumber) {this._discount = aNumber;}
```

“ ”

“ ”

240

302

132

237

**class ProductionPlan...**

```
get production() {return this._production;}
applyAdjustment(anAdjustment) {
 this._adjustments.push(anAdjustment);
 this._production += anAdjustment.amount;
}
```

——  production plan  adjustment
——

“  ”  ——  302

## class ProductionPlan...

```
get production() {
 assert(this._production === this.calculatedProduction);
 return this._production;
}

get calculatedProduction() {
 return this._adjustments
  .reduce((sum, a) => sum + a.amount, 0);
}
```

## class ProductionPlan...

```
get production() {
  assert(this._production === this.calculatedProduction);
  return this.calculatedProduction;
}
```

115  production

## class ProductionPlan...

```
get production() {
  return this._adjustments
    .reduce((sum, a) => sum + a.amount, 0);
}
```

237

## class ProductionPlan...

```
  applyAdjustment(anAdjustment) {
  this._adjustments.push(anAdjustment);
  this._production += anAdjustment.amount;
}
```

production

## class ProductionPlan...

```
  constructor (production) {
 this._production = production;
 this._adjustments = [];
}
get production() {return this._production;}
applyAdjustment(anAdjustment) {
 this._adjustments.push(anAdjustment);
 this._production += anAdjustment.amount;
}
```

302 production 0

240

```
constructor (production) {
 this._initialProduction = production;
 this._productionAccumulator = 0;
 this._adjustments = [];
}
get production() {
 return this._initialProduction + this._productionAccumulator;
}
```

302

## class ProductionPlan...

```
get production() {
 assert(this._productionAccumulator === this.calculatedProductionAccumulator);
 return this._initialProduction + this._productionAccumulator;
}

get calculatedProductionAccumulator() {
 return this._adjustments
  .reduce((sum, a) => sum + a.amount, 0);
}
```

calculatedProduction Accumulator

# 9.4 Change Reference to Value

256

```
class Product {
applyDiscount(arg) {this._price.amount -= arg;}


class Product {
applyDiscount(arg) {
  this._price = new Money(this._price.amount - arg, this._price.currency);
}
```

[mf-vo]

331

“ ” Person “ ” Telephone Number

## class Person...

```
  constructor() {
  constructor() {
 this._telephoneNumber = new TelephoneNumber();
}

get officeAreaCode()  {return this._telephoneNumber.areaCode;}
set officeAreaCode(arg) {this._telephoneNumber.areaCode = arg;}
get officeNumber()  {return this._telephoneNumber.number;}
set officeNumber(arg) {this._telephoneNumber.number = arg;}
```

## class TelephoneNumber...

```
  get areaCode()    {return this._areaCode;}
set areaCode(arg) {this._areaCode = arg;}

get number()    {return this._number;}
set number(arg) {this._number = arg;}
```

182 Person

TelephoneNumber 331 331 124

## class TelephoneNumber...

```
  constructor(areaCode, number) {
  this._areaCode = areaCode;
  this._number = number;
}
```

“ ” area code

## class Person...

```
  get officeAreaCode()    {return this._telephoneNumber.areaCode;}
set officeAreaCode(arg) {
 this._telephoneNumber = new TelephoneNumber(arg, this.officeNumber);
}
get officeNumber()   {return this._telephoneNumber.number;}
set officeNumber(arg) {this._telephoneNumber.number = arg;}
```

## class Person...

```
  get officeAreaCode()    {return this._telephoneNumber.areaCode;}
set officeAreaCode(arg) {
  this._telephoneNumber = new TelephoneNumber(arg, this.officeNumber);
}
get officeNumber()    {return this._telephoneNumber.number;}
set officeNumber(arg) {
 this._telephoneNumber = new TelephoneNumber(this.officeAreaCode, arg);
}
```

TelephoneNumber JavaScript

“ ” “ ” equals

### class TelephoneNumber...

```
  equals(other) {
 if (!(other instanceof TelephoneNumber)) return false;
 return this.areaCode === other.areaCode &amp;&amp;
  this.number === other.number;
}
```

```
it("telephone equals", function () {
  assert(
    new TelephoneNumber("312", "555-0142").equals(
      new TelephoneNumber("312", "555-0142")
    )
  );
});
```

Ruby == Java Object.equals()
Java Object.hashCode()

TelephoneNumber 331
null

## 9.5 Change Value to Reference

252

```
let customer = new Customer(customerData);

let customer = customerRepository.get(customerData.id);
```

“ ” Order JSON customer ID Customer

## class Order...

```
  constructor(data) {
  this._number = data.number;
  this._customer = new Customer(data.customer);
  // load other data
}
get customer() {return this._customer;}
```

## class Customer...

```
  constructor(id) {
  this._id = id;
}
get id() {return this._id;}
```

Customer 5 ID 123 5 Customer

Customer 5 ——

Customer

Customer [mf-repos]

```
let _repositoryData;

export function initialize() {
  _repositoryData = {};
  _repositoryData.customers = new Map();
}

export function registerCustomer(id) {
  if (!_repositoryData.customers.has(id))
    _repositoryData.customers.set(id, new Customer(id));
  return findCustomer(id);
}

export function findCustomer(id) {
  return _repositoryData.customers.get(id);
}
```

ID ID Customer Order

Order Customer ID

## class Order...

```
  constructor(data) {
 this._number = data.number;
 this._customer = registerCustomer(data.customer);
 // load other data
}
get customer() {return this._customer;}
```

Order Customer Customer

Order Customer Order ID Customer

# 10

260 263
266 “ ” switch 272

null 289
302

## 10.1 Decompose Conditional

```
if (!aDate.isBefore(plan.summerStart) &amp;&amp; !aDate.isAfter(plan.summerEnd)
 charge = quantity * plan.summerRate;
else
 charge = quantity * plan.regularRate + plan.regularServiceCharge;


if (summer())
 charge = summerCharge();
else
 charge = regularCharge();
```

106

106

= ×

```
if (!aDate.isBefore(plan.summerStart) &amp;&amp; !aDate.isAfter(plan.summerEnd)
 charge = quantity * plan.summerRate;
else
 charge = quantity * plan.regularRate + plan.regularServiceCharge;
```

```
if (summer())
 charge = quantity * plan.summerRate;
else
 charge = quantity * plan.regularRate + plan.regularServiceCharge;

function summer() {
 return !aDate.isBefore(plan.summerStart) &amp;amp;&amp;amp; !aDate.isAfter(plan.summer
}
```

```
if (summer())
 charge = summerCharge();
else
 charge = quantity * plan.regularRate + plan.regularServiceCharge;

function summer() {
 return !aDate.isBefore(plan.summerStart) &amp;amp;&amp;amp; !aDate.isAfter(plan.summer
}
function summerCharge() {
 return quantity * plan.summerRate;
}
```

```
if (summer())
 charge = summerCharge();
else
 charge = regularCharge();

function summer() {
 return !aDate.isBefore(plan.summerStart) &amp;amp;&amp;amp; !aDate.isAfter(plan.summer
}
function summerCharge() {
 return quantity * plan.summerRate;
}
function regularCharge() {
 return quantity * plan.regularRate + plan.regularServiceCharge;
}
```

```
  charge = summer() ? summerCharge() : regularCharge();

function summer() {
 return !aDate.isBefore(plan.summerStart) &amp;amp;&amp;amp; !aDate.isAfter(plan.summer
}
function summerCharge() {
 return quantity * plan.summerRate;
}
function regularCharge() {
 return quantity * plan.regularRate + plan.regularServiceCharge;
}
```

# 10.2 Consolidate Conditional Expression

```
if (anEmployee.seniority < 2) return 0;
if (anEmployee.monthsDisabled > 12) return 0;
if (anEmployee.isPartTime) return 0;


if (isNotEligibleForDisability()) return 0;

function isNotEligibleForDisability() {
 return ((anEmployee.seniority < 2)
     || (anEmployee.monthsDisabled > 12)
     || (anEmployee.isPartTime));
}
```

“ ” “ ”

“ ”

“ ” 106

“ ” “ ”

306

if

106

```
function disabilityAmount(anEmployee) {
 if (anEmployee.seniority < 2) return 0;
 if (anEmployee.monthsDisabled > 12) return 0;
 if (anEmployee.isPartTime) return 0;
 // compute the disability amount
```

```
function disabilityAmount(anEmployee) {
 if ((anEmployee.seniority < 2)
   || (anEmployee.monthsDisabled > 12)) return 0;
 if (anEmployee.isPartTime) return 0;
 // compute the disability amount
```

```
function disabilityAmount(anEmployee) {
 if ((anEmployee.seniority < 2)
   || (anEmployee.monthsDisabled > 12)
   || (anEmployee.isPartTime)) return 0;
 // compute the disability amount
```

106

```
function disabilityAmount(anEmployee) {
 if (isNotEligableForDisability()) return 0;
 // compute the disability amount

function isNotEligableForDisability() {
 return ((anEmployee.seniority < 2)
     || (anEmployee.monthsDisabled > 12)
     || (anEmployee.isPartTime));
}
```

if

```
if (anEmployee.onVacation)
 if (anEmployee.seniority > 10)
  return 1;
return 0.5;
```

```
if ((anEmployee.onVacation)
  &amp;amp;&amp;amp; (anEmployee.seniority > 10)) return 1;
return 0.5;
```

106

## 10.3 Replace Nested Conditional with Guard Clauses

```
function getPayAmount() {
  let result;
  if (isDead) result = deadAmount();
  else {
    if (isSeparated) result = separatedAmount();
    else {
      if (isRetired) result = retiredAmount();
      else result = normalPayAmount();
    }
  }
  return result;
}

function getPayAmount() {
  if (isDead) return deadAmount();
  if (isSeparated) return separatedAmount();
  if (isRetired) return retiredAmount();
  return normalPayAmount();
}
```

if...else...

“ ” guard clauses

if-then-else if else

“ ”

“ ”

“ ”

263

employee “ ”

```
function payAmount(employee) {
 let result;
 if(employee.isSeparated) {
  result = {amount: 0, reasonCode:"SEP"};
 }
 else {
  if (employee.isRetired) {
   result = {amount: 0, reasonCode: "RET"};
  }
  else {
   // logic to compute amount
   lorem.ipsum(dolor.sitAmet);1
   consectetur(adipiscing).elit();
   sed.do.eiusmod = tempor.incididunt.ut(labore) &amp;&amp; dolore(magna.aliqua
   ut.enim.ad(minim.veniam);
   result = someFinalComputation();
  }
 }
 return result;
}
```

```
function payAmount(employee) {
 let result;
 if (employee.isSeparated) return {amount: 0, reasonCode: "SEP"};
 if (employee.isRetired) {
  result = {amount: 0, reasonCode: "RET"};
 }
 else {
  // logic to compute amount
  lorem.ipsum(dolor.sitAmet);
  consectetur(adipiscing).elit();
  sed.do.eiusmod = tempor.incididunt.ut(labore) &amp;&amp; dolore(magna.aliqua
  ut.enim.ad(minim.veniam);
  result = someFinalComputation();
 }
 return result;
}
```

```
function payAmount(employee) {
 let result;
 if (employee.isSeparated) return {amount: 0, reasonCode: "SEP"};
 if (employee.isRetired)   return {amount: 0, reasonCode: "RET"};
 // logic to compute amount
 lorem.ipsum(dolor.sitAmet);
 consectetur(adipiscing).elit();
 sed.do.eiusmod = tempor.incididunt.ut(labore) &amp;&amp; dolore(magna.aliqua);
 ut.enim.ad(minim.veniam);
 result = someFinalComputation();
 return result;
}
```

result

```
function payAmount(employee) {
 let result;
 if (employee.isSeparated) return {amount: 0, reasonCode: "SEP"};
 if (employee.isRetired)   return {amount: 0, reasonCode: "RET"};
 // logic to compute amount
 lorem.ipsum(dolor.sitAmet);
 consectetur(adipiscing).elit();
 sed.do.eiusmod = tempor.incididunt.ut(labore) &amp;&amp; dolore(magna.aliqua);
 ut.enim.ad(minim.veniam);
 return someFinalComputation();
}
```

1 Joshua Kerievsky

```
function adjustedCapital(anInstrument) {
 let result = 0;
 if (anInstrument.capital > 0) {
  if (anInstrument.interestRate > 0 &amp;amp;&amp;amp; anInstrument.duration > 0) {
   result = (anInstrument.income / anInstrument.duration) * anInstrument.adjust
  }
 }
 return result;
}
```

```
function adjustedCapital(anInstrument) {
 let result = 0;
 if (anInstrument.capital <= 0) return result;
 if (anInstrument.interestRate > 0 &amp;amp;&amp;amp; anInstrument.duration > 0) {
  result = (anInstrument.income / anInstrument.duration) * anInstrument.adjustr
 }
 return result;
}
```

```
function adjustedCapital(anInstrument) {
 let result = 0;
 if (anInstrument.capital <= 0) return result;
 if (!(anInstrument.interestRate > 0 &amp;amp;&amp;amp; anInstrument.duration > 0)) ret
 result = (anInstrument.income / anInstrument.duration) * anInstrument.adjustme
 return result;
}
```

```
 function adjustedCapital(anInstrument) {
let result = 0;
if (anInstrument.capital <= 0) return result;
if (anInstrument.interestRate <= 0 || anInstrument.duration <= 0) return resul
result = (anInstrument.income / anInstrument.duration) * anInstrument.adjustme
return result;
}
```

263

```
function adjustedCapital(anInstrument) {
 let result = 0;
 if (   anInstrument.capital      <= 0
   || anInstrument.interestRate <= 0
   || anInstrument.duration     <= 0) return result;
 result = (anInstrument.income / anInstrument.duration) * anInstrument.adjustme
 return result;
}
```

result 0

```
function adjustedCapital(anInstrument) {
 if (   anInstrument.capital     <= 0
   || anInstrument.interestRate <= 0

   || anInstrument.duration   <= 0) return 0;
 return (anInstrument.income / anInstrument.duration) * anInstrument.adjustment
}
```

1 “lorem.ipsum……” ——

# 10.4 Replace Conditional with Polymorphism

```
switch (bird.type) {
 case 'EuropeanSwallow':
  return "average";
 case 'AfricanSwallow':
  return (bird.numberOfCoconuts > 2) ? "tired" : "average";
 case 'NorwegianBlueParrot':
  return (bird.voltage > 100) ? "scorched" : "beautiful";
 default:
  return "unknown";


class EuropeanSwallow {
 get plumage() {
  return "average";
 }
class AfricanSwallow {
 get plumage() {
   return (this.numberOfCoconuts > 2) ? "tired" : "average";
 }
class NorwegianBlueParrot {
 get plumage() {
   return (this.voltage > 100) ? "scorched" : "beautiful";
}
```

switch switch

——

if/else switch/case

106

abstract

```
function plumages(birds) {
 return new Map(birds.map(b => [b.name, plumage(b)]));
}
function speeds(birds) {
 return new Map(birds.map(b => [b.name, airSpeedVelocity(b)]));
}

function plumage(bird) {
 switch (bird.type) {
 case 'EuropeanSwallow':
  return "average";
 case 'AfricanSwallow':
  return (bird.numberOfCoconuts > 2) ? "tired" : "average";
 case 'NorwegianBlueParrot':
  return (bird.voltage > 100) ? "scorched" : "beautiful";
 default:
  return "unknown";
 }
}

function airSpeedVelocity(bird) {
 switch (bird.type) {
 case 'EuropeanSwallow':
  return 35;
 case 'AfricanSwallow':
  return 40 - 2 * bird.numberOfCoconuts;
 case 'NorwegianBlueParrot':
  return (bird.isNailed) ? 0 : 10 + bird.voltage / 10;
 default:
  return null;
 }
}
```

“ ”

airSpeedVelocity plumage 144

```
function plumage(bird) {
 return new Bird(bird).plumage;
}

function airSpeedVelocity(bird) {
 return new Bird(bird).airSpeedVelocity;
}
class Bird {
 constructor(birdObject) {
  Object.assign(this, birdObject);
 }
 get plumage() {
  switch (this.type) {
  case 'EuropeanSwallow':
   return "average";
  case 'AfricanSwallow':
   return (this.numberOfCoconuts > 2) ? "tired" : "average";
  case 'NorwegianBlueParrot':
   return (this.voltage > 100) ? "scorched" : "beautiful";
  default:
   return "unknown";
  }
 }
 get airSpeedVelocity() {
  switch (this.type) {
  case 'EuropeanSwallow':
   return 35;
  case 'AfricanSwallow':
   return 40 - 2 * this.numberOfCoconuts;
  case 'NorwegianBlueParrot':
   return (this.isNailed) ? 0 : 10 + this.voltage / 10;
  default:
   return null;
  }
 }
}
```

```
function plumage(bird) {
  return createBird(bird).plumage;
}

function airSpeedVelocity(bird) {
  return createBird(bird).airSpeedVelocity;
}

function createBird(bird) {
  switch (bird.type) {
    case "EuropeanSwallow":
      return new EuropeanSwallow(bird);
    case "AfricanSwallow":
      return new AfricanSwallow(bird);
    case "NorweigianBlueParrot":
      return new NorwegianBlueParrot(bird);
    default:
      return new Bird(bird);
  }
}
class EuropeanSwallow extends Bird {}

class AfricanSwallow extends Bird {}

class NorwegianBlueParrot extends Bird {}
```

plumage switch

## class EuropeanSwallow...

```
get plumage() {
  return "average";
}
```

## class Bird...

```
get plumage() {
 switch (this.type) {
 case 'EuropeanSwallow':
  throw "oops";
 case 'AfricanSwallow':
  return (this.numberOfCoconuts > 2) ? "tired" : "average";
 case 'NorwegianBlueParrot':
  return (this.voltage > 100) ? "scorched" : "beautiful";
 default:
  return "unknown";
 }
}
```

## class AfricanSwallow...

```
get plumage() {
   return (this.numberOfCoconuts > 2) ? "tired" : "average";
}
```

Norwegian Blue

## class NorwegianBlueParrot...

```
get plumage() {
   return (this.voltage >100) ? "scorched" : "beautiful";
}
```

## class Bird...

```
get plumage() {
  return "unknown";
}
```

airSpeedVelocity airSpeedVelocity plumage

```
function plumages(birds) {
 return new Map(birds
         .map(b => createBird(b))
         .map(bird => [bird.name, bird.plumage]));
}
function speeds(birds) {
 return new Map(birds
         .map(b => createBird(b))
         .map(bird => [bird.name, bird.airSpeedVelocity]));
}

function createBird(bird) {
 switch (bird.type) {
 case 'EuropeanSwallow':
  return new EuropeanSwallow(bird);
 case 'AfricanSwallow':
  return new AfricanSwallow(bird);
 case 'NorwegianBlueParrot':
  return new NorwegianBlueParrot(bird);
 default:
  return new Bird(bird);
 }
}

class Bird {
 constructor(birdObject) {
  Object.assign(this, birdObject);
 }
 get plumage() {
  return "unknown";
 }
 get airSpeedVelocity() {
  return null;
 }
}
class EuropeanSwallow extends Bird {
 get plumage() {
  return "average";
 }
 get airSpeedVelocity() {
  return 35;
 }
}
class AfricanSwallow extends Bird {
 get plumage() {
  return (this.numberOfCoconuts > 2) ? "tired" : "average";
 }
 get airSpeedVelocity() {
  return 40 - 2 * this.numberOfCoconuts;
 }
}
```

```
class NorwegianBlueParrot extends Bird {
 get plumage() {
  return (this.voltage > 100) ? "scorched" : "beautiful";
 }
 get airSpeedVelocity() {
  return (this.isNailed) ? 0 : 10 + this.voltage / 10;
 }
}
```

Bird JavaScript

“ ” “ ”

“A” “B”

```
function rating(voyage, history) {
 const vpf = voyageProfitFactor(voyage, history);
 const vr = voyageRisk(voyage);
 const chr = captainHistoryRisk(voyage, history);
 if (vpf * 3 > (vr + chr * 2)) return "A";
 else return "B";
}
function voyageRisk(voyage) {
 let result = 1;
 if (voyage.length > 4) result += 2;
 if (voyage.length > 8) result += voyage.length - 8;
 if (["china", "east-indies"].includes(voyage.zone)) result += 4;
 return Math.max(result, 0);
}
function captainHistoryRisk(voyage, history) {
 let result = 1;
 if (history.length < 5) result += 4;
 result += history.filter(v => v.profit < 0).length;
 if (voyage.zone === "china" &amp;&amp; hasChina(history)) result -= 2;
 return Math.max(result, 0);
}
function hasChina(history) {
 return history.some(v => "china" === v.zone);
}
function voyageProfitFactor(voyage, history) {
 let result = 2;
 if (voyage.zone === "china") result += 1;
 if (voyage.zone === "east-indies") result += 1;
 if (voyage.zone === "china" &amp;&amp; hasChina(history)) {
  result += 3;
  if (history.length > 10) result += 1;
  if (voyage.length > 12) result += 1;
  if (voyage.length > 18) result -= 1;
 }
 else {
  if (history.length > 8) result += 1;
  if (voyage.length > 14) result -= 1;
 }
 return result;
}
```

voyageRisk captainHistoryRisk voyageProfitFactor rating 3

```
const voyage = { zone: "west-indies", length: 10 };
const history = [
  { zone: "east-indies", profit: 5 },
  { zone: "west-indies", profit: 15 },
  { zone: "china", profit: -2 },
  { zone: "west-africa", profit: 7 },
];

const myRating = rating(voyage, history);
```

“ ” “ ”

```
function rating(voyage, history) {
 const vpf = voyageProfitFactor(voyage, history);
 const vr = voyageRisk(voyage);
 const chr = captainHistoryRisk(voyage, history);
 if (vpf * 3 > (vr + chr * 2)) return "A";
 else return "B";
}
function voyageRisk(voyage) {
 let result = 1;
 if (voyage.length > 4) result += 2;
 if (voyage.length > 8) result += voyage.length - 8;
 if (["china", "east-indies"].includes(voyage.zone)) result += 4;
 return Math.max(result, 0);
}
function captainHistoryRisk(voyage, history) {
 let result = 1;
 if (history.length < 5) result += 4;
 result += history.filter(v => v.profit < 0).length;
 if (voyage.zone === "china" &amp;&amp; hasChina(history)) result -= 2;
 return Math.max(result, 0);
}
function hasChina(history) {
 return history.some(v => "china" === v.zone);
}
function voyageProfitFactor(voyage, history) {
 let result = 2;
 if (voyage.zone === "china") result += 1;
 if (voyage.zone === "east-indies") result += 1;
 if (voyage.zone === "china" &amp;&amp; hasChina(history)) {
  result += 3;
  if (history.length > 10) result += 1;
  if (voyage.length > 12) result += 1;
  if (voyage.length > 18) result -= 1;
 }
 else {
  if (history.length > 8) result += 1;
  if (voyage.length > 14) result -= 1;
 }
 return result;
}
```

“ ” ____ “ ”

```
function rating(voyage, history) {
 return new Rating(voyage, history).value;
}

class Rating {
 constructor(voyage, history) {
  this.voyage = voyage;
  this.history = history;
 }
 get value() {
  const vpf = this.voyageProfitFactor;
  const vr = this.voyageRisk;
  const chr = this.captainHistoryRisk;
  if (vpf * 3 > (vr + chr * 2)) return "A";
  else return "B";
 }
 get voyageRisk() {
  let result = 1;
  if (this.voyage.length > 4) result += 2;
  if (this.voyage.length > 8) result += this.voyage.length - 8;
  if (["china", "east-indies"].includes(this.voyage.zone)) result += 4;
  return Math.max(result, 0);
 }
 get captainHistoryRisk() {
  let result = 1;
  if (this.history.length < 5) result += 4;
  result += this.history.filter(v => v.profit < 0).length;
  if (this.voyage.zone === "china" &amp;&amp; this.hasChinaHistory) result -= 2
  return Math.max(result, 0);
 }
 get voyageProfitFactor() {
  let result = 2;

  if (this.voyage.zone === "china") result += 1;
  if (this.voyage.zone === "east-indies") result += 1;
  if (this.voyage.zone === "china" &amp;&amp; this.hasChinaHistory) {
   result += 3;
   if (this.history.length > 10) result += 1;
   if (this.voyage.length > 12) result += 1;
   if (this.voyage.length > 18) result -= 1;
  }
  else {
   if (this.history.length > 8) result += 1;
   if (this.voyage.length > 14) result -= 1;
  }
  return result;
 }
 get hasChinaHistory() {
  return this.history.some(v => "china" === v.zone);
 }
}
```

```
class ExperiencedChinaRating extends Rating {}
```

```
function createRating(voyage, history) {
 if (voyage.zone === "china" &amp;amp;&amp;amp; history.some(v => "china" === v.zone))
  return new ExperiencedChinaRating(voyage, history);
 else return new Rating(voyage, history);
}
```

rating

```
function rating(voyage, history) {
  return createRating(voyage, history).value;
}
```

captainHistoryRisk

## class Rating...

```
get captainHistoryRisk() {
 let result = 1;
 if (this.history.length < 5) result += 4;
 result += this.history.filter(v => v.profit < 0).length;
 if (this.voyage.zone === "china" &amp;amp;&amp;amp; this.hasChinaHistory) result -= 2;
 return Math.max(result, 0);
}
```

class ExperiencedChinaRating

```
get captainHistoryRisk() {
  const result = super.captainHistoryRisk - 2;
  return Math.max(result, 0);
}
```

## class Rating...

```
get captainHistoryRisk() {
 let result = 1;
 if (this.history.length < 5) result += 4;
 result += this.history.filter(v => v.profit < 0).length;
 if (this.voyage.zone === "china" &amp;amp; this.hasChinaHistory) result -= 2;
 return Math.max(result, 0);
}
```

voyageProfitFactor

## class Rating...

```
get voyageProfitFactor() {
 let result = 2;

 if (this.voyage.zone === "china") result += 1;
 if (this.voyage.zone === "east-indies") result += 1;
 if (this.voyage.zone === "china" &amp;amp; this.hasChinaHistory) {
  result += 3;
  if (this.history.length > 10) result += 1;
  if (this.voyage.length > 12) result += 1;
  if (this.voyage.length > 18) result -= 1;
 }
 else {
  if (this.history.length > 8) result += 1;
  if (this.voyage.length > 14) result -= 1;
 }
 return result;
}
```

106

## class Rating...

```
get voyageProfitFactor() {
 let result = 2;

 if (this.voyage.zone === "china") result += 1;
 if (this.voyage.zone === "east-indies") result += 1;
 result += this.voyageAndHistoryLengthFactor;
 return result;
}
get voyageAndHistoryLengthFactor() {
 let result = 0;
 if (this.voyage.zone === "china" &amp;amp; this.hasChinaHistory) {
  result += 3;
  if (this.history.length > 10) result += 1;
  if (this.voyage.length > 12) result += 1;
  if (this.voyage.length > 18) result -= 1;
 }
 else {
  if (this.history.length > 8) result += 1;
  if (this.voyage.length > 14) result -= 1;
 }
 return result;
}
```

“And”

## class Rating...

```
get voyageAndHistoryLengthFactor() {
 let result = 0;
 if (this.history.length > 8) result += 1;
 if (this.voyage.length > 14) result -= 1;
 return result;
}
```

## class ExperiencedChinaRating...

```
get voyageAndHistoryLengthFactor() {
 let result = 0;
 result += 3;
 if (this.history.length > 10) result += 1;
 if (this.voyage.length > 12) result += 1;
 if (this.voyage.length > 18) result -= 1;
 return result;
}
```

———

“ ” ____ ____

“And” 106 “ ” history length

## class Rating...

```
get voyageAndHistoryLengthFactor() {
 let result = 0;
 result += this.historyLengthFactor;
 if (this.voyage.length > 14) result -= 1;
 return result;
}
get historyLengthFactor() {
 return (this.history.length > 8) ? 1 : 0;
}
```

## class ExperiencedChinaRating...

```
get voyageAndHistoryLengthFactor() {
 let result = 0;
 result += 3;
 result += this.historyLengthFactor;
 if (this.voyage.length > 12) result += 1;
 if (this.voyage.length > 18) result -= 1;
 return result;
}
get historyLengthFactor() {
 return (this.history.length > 10) ? 1 : 0;
}
```

217

## class Rating...

```
get voyageProfitFactor() {
 let result = 2;
 if (this.voyage.zone === "china") result += 1;
 if (this.voyage.zone === "east-indies") result += 1;
 result += this.historyLengthFactor;
 result += this.voyageAndHistoryLengthFactor;
 return result;
}

get voyageAndHistoryLengthFactor() {
 let result = 0;
 result += this.historyLengthFactor;
 if (this.voyage.length > 14) result -= 1;
 return result;
}
```

## class ExperiencedChinaRating...

```
get voyageAndHistoryLengthFactor() {
 let result = 0;
 result += 3;
 result += this.historyLengthFactor;
 if (this.voyage.length > 12) result += 1;
 if (this.voyage.length > 18) result -= 1;
 return result;
}
```

124

## class Rating...

```
get voyageProfitFactor() {
 let result = 2;
 if (this.voyage.zone === "china") result += 1;
 if (this.voyage.zone === "east-indies") result += 1;
 result += this.historyLengthFactor;
 result += this.voyageLengthFactor;
 return result;
}

get voyageLengthFactor() {
 return (this.voyage.length > 14) ? - 1: 0;
}
```

voyageLengthFactor

## class ExperiencedChinaRating...

```
get voyageLengthFactor() {
 let result = 0;
 result += 3;
 if (this.voyage.length > 12) result += 1;
 if (this.voyage.length > 18) result -= 1;
 return result;
}
```

“ ” voyage length 3 3

## class ExperiencedChinaRating...

```
get voyageProfitFactor() {
  return super.voyageProfitFactor + 3;
}

get voyageLengthFactor() {
  let result = 0;
  result += 3;
  if (this.voyage.length > 12) result += 1;
  if (this.voyage.length > 18) result -= 1;
  return result;
}
```

Rating “ ”

```
class Rating {
 constructor(voyage, history) {
  this.voyage = voyage;
  this.history = history;
 }
 get value() {
  const vpf = this.voyageProfitFactor;
  const vr = this.voyageRisk;
  const chr = this.captainHistoryRisk;
  if (vpf * 3 > (vr + chr * 2)) return "A";
  else return "B";
 }
 get voyageRisk() {
  let result = 1;
  if (this.voyage.length > 4) result += 2;
  if (this.voyage.length > 8) result += this.voyage.length - 8;
  if (["china", "east-indies"].includes(this.voyage.zone)) result += 4;
  return Math.max(result, 0);
 }
 get captainHistoryRisk() {
  let result = 1;
  if (this.history.length < 5) result += 4;
  result += this.history.filter(v => v.profit < 0).length;
  return Math.max(result, 0);
 }
 get voyageProfitFactor() {
  let result = 2;
  if (this.voyage.zone === "china") result += 1;
  if (this.voyage.zone === "east-indies") result += 1;
  result += this.historyLengthFactor;
  result += this.voyageLengthFactor;
  return result;
 }
 get voyageLengthFactor() {
  return (this.voyage.length > 14) ? - 1: 0;
 }
 get historyLengthFactor() {
  return (this.history.length > 8) ? 1 : 0;
 }
}
```

“ ”

```
class ExperiencedChinaRating extends Rating {
 get captainHistoryRisk() {
  const result = super.captainHistoryRisk - 2;
  return Math.max(result, 0);
 }
 get voyageLengthFactor() {
  let result = 0;
  if (this.voyage.length > 12) result += 1;
  if (this.voyage.length > 18) result -= 1;
  return result;
 }
 get historyLengthFactor() {
  return (this.history.length > 10) ? 1 : 0;
 }
 get voyageProfitFactor() {
  return super.voyageProfitFactor + 3;
 }
}
```

## 10.5 Introduce Special Case

Null Introduce Null Object

```
if (aCustomer === "unknown") customerName = "occupant";


class UnknownCustomer {
  get name() {return "occupant";}
```

“ ” Special Case

literal object

null “Null ” Null Object —— Null

false

true

“ ” 106

144 149

literal record

115

site

## class Site...

```
get customer() {return this._customer;}
```

“ ” Customer 3

## class Customer...

```
get name()           {...}
get billingPlan()    {...}
set billingPlan(arg) {...}
get paymentHistory() {...}
```

customer

"unknown" Site “ ”

### 1...

```
const aCustomer = site.customer;
// ... lots of intervening code ...
let customerName;
if (aCustomer === "unknown") customerName = "occupant";
else customerName = aCustomer.name;
```

### 2...

```
const plan =
  aCustomer === "unknown" ? registry.billingPlans.basic : aCustomer.billingPlar
```

### 3...

```
if (aCustomer !== "unknown") aCustomer.billingPlan = newPlan;
```

### 4...

```
const weeksDelinquent =
  aCustomer === "unknown"
    ? 0
    : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Site “ ” "occupant"
Special Case Object

Customer “ ”

## class Customer...

```
get isUnknown() {return false;}
```

“ ”

```
class UnknownCustomer {
  get isUnknown() {
    return true;
  }
}
```

UnknownCustomer Customer JavaScript

"unknown" "unknown" isUnknown
Customer UnknownCustomer
"unknown" "unknown" isUnknown ——

“ ” 106

```
function isUnknown(arg) {
  if (!(arg instanceof Customer || arg === "unknown"))
    throw new Error(`investigate bad value: <${arg}>`);
  return arg === "unknown";
}
```

### 1...

```
let customerName;
if (isUnknown(aCustomer)) customerName = "occupant";
else customerName = aCustomer.name;
```

### 2...

```
const plan = isUnknown(aCustomer)
  ? registry.billingPlans.basic
  : aCustomer.billingPlan;
```

### 3...

```
if (!isUnknown(aCustomer)) aCustomer.billingPlan = newPlan;
```

### 4...

```
const weeksDelinquent = isUnknown(aCustomer)
  ? 0
  : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

isUnknown Site UnknownCustomer

## class Site...

```
get customer() {
  return (this._customer === "unknown") ? new UnknownCustomer() : this._custome
}
```

isUnknown "unknown"

### 1...

```
function isUnknown(arg) {
  if (!(arg instanceof Customer || arg instanceof UnknownCustomer))
    throw new Error(`investigate bad value: <${arg}>`);
  return arg.isUnknown;
}
```

144

"occupant"

### 1...

```
let customerName;
if (isUnknown(aCustomer)) customerName = "occupant";
else customerName = aCustomer.name;
```

UnknownCustomer

## class UnknownCustomer...

```
get name() {return "occupant";}
```

### 1...

```
const customerName = aCustomer.name;
```

123 customerName

“ ” billingPlan

### 2...

```
const plan = isUnknown(aCustomer)
  ? registry.billingPlans.basic
  : aCustomer.billingPlan;
```

### 3...

```
if (!isUnknown(aCustomer)) aCustomer.billingPlan = newPlan;
```

name —— UnknownCustomer

## class UnknownCustomer...

```
get billingPlan()  {return registry.billingPlans.basic;}
set billingPlan(arg) { /* ignore */ }
```

...

```
const plan = aCustomer.billingPlan;
```

...

```
aCustomer.billingPlan = newPlan;
```

...

```
const weeksDelinquent = isUnknown(aCustomer)
  ? 0
  : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

“ ” NullPaymentHistory

## class UnknownCustomer...

```
get paymentHistory() {return new NullPaymentHistory();}
```

## class NullPaymentHistory...

```
get weeksDelinquentInLastYear() {return 0;}
```

...

```
const weeksDelinquent = aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

—— 23 "occupant"

...

```
const name = !isUnknown(aCustomer) ? aCustomer.name : "unknown occupant";
```

aCustomer isUnknown isUnknown

115

...

```
const name = aCustomer.isUnknown ? "unknown occupant" : aCustomer.name;
```

isUnknown 237

Customer

literal object

—— Customer

## class Site...

```
get customer() {return this._customer;}
```

## class Customer...

```
get name()           {...}
get billingPlan()    {...}
set billingPlan(arg) {...}
get paymentHistory() {...}
```

### 1...

```
const aCustomer = site.customer;
// ... lots of intervening code ...
let customerName;
if (aCustomer === "unknown") customerName = "occupant";
else customerName = aCustomer.name;
```

### 2...

```
const plan =
  aCustomer === "unknown" ? registry.billingPlans.basic : aCustomer.billingPlar
```

### 3...

```
const weeksDelinquent =
  aCustomer === "unknown"
    ? 0
    : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Customer isUnknown

## class Customer...

```
get isUnknown() {return false;}
```

...

```
function createUnknownCustomer() {
  return {
    isUnknown: true,
  };
}
```

106

```
function isUnknown(arg) {
  return arg === "unknown";
}
```

1...

```
let customerName;
if (isUnknown(aCustomer)) customerName = "occupant";
else customerName = aCustomer.name;
```

2...

```
const plan = isUnknown(aCustomer)
  ? registry.billingPlans.basic
  : aCustomer.billingPlan;
```

3...

```
const weeksDelinquent = isUnknown(aCustomer)
  ? 0
  : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Site isUnknown

## class Site...

```
get customer() {
  return (this._customer === "unknown") ? createUnknownCustomer() : this._custo
}
```

...

```
function isUnknown(arg) {
  return arg.isUnknown;
}
```

“ ” “ ”

```
function createUnknownCustomer() {
  return {
    isUnknown: true,
    name: "occupant",
  };
}
```

### 1...

```
const customerName = aCustomer.name;
```

“ ”

```
function createUnknownCustomer() {
  return {
    isUnknown: true,
    name: "occupant",
    billingPlan: registry.billingPlans.basic,
  };
}
```

### 2...

```
const plan = aCustomer.billingPlan;
```

```
function createUnknownCustomer() {
  return {
    isUnknown: true,
    name: "occupant",
    billingPlan: registry.billingPlans.basic,
    paymentHistory: {
      weeksDelinquentInLastYear: 0,
    },
  };
}
```

### 3...

```
const weeksDelinquent = aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Object.freeze

```
{
 name: "Acme Boston",
 location: "Malden MA",
 // more site details
 customer: {
  name: "Acme Industries",
  billingPlan: "plan-451",
  paymentHistory: {
   weeksDelinquentInLastYear: 7
   //more
  },
  // more
 }
}
```

customer "unknown"

```
{
name: "Warehouse Unit 15",
location: "Malden MA",
// more site details
customer: "unknown",
}
```

“ ”

**1...**

```
const site = acquireSiteData();
const aCustomer = site.customer;
// ... lots of intervening code ...
let customerName;
if (aCustomer === "unknown") customerName = "occupant";
else customerName = aCustomer.name;
```

**2...**

```
const plan =
  aCustomer === "unknown" ? registry.billingPlans.basic : aCustomer.billingPlar
```

### 3...

```
const weeksDelinquent =
  aCustomer === "unknown"
    ? 0
    : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Site

### 1...

```
const rawSite = acquireSiteData();
const site = enrichSite(rawSite);
const aCustomer = site.customer;
// ... lots of intervening code ...
let customerName;
if (aCustomer === "unknown") customerName = "occupant";
else customerName = aCustomer.name;

function enrichSite(inputSite) {
  return _.cloneDeep(inputSite);
}
```

“ ” 106

```
function isUnknown(aCustomer) {
  return aCustomer === "unknown";
}
```

### 1...

```
const rawSite = acquireSiteData();
const site = enrichSite(rawSite);
const aCustomer = site.customer;
// ... lots of intervening code ...
let customerName;
if (isUnknown(aCustomer)) customerName = "occupant";
else customerName = aCustomer.name;
```

### 2...

```
const plan = isUnknown(aCustomer)
  ? registry.billingPlans.basic
  : aCustomer.billingPlan;
```

### 3...

```
const weeksDelinquent = isUnknown(aCustomer)
  ? 0
  : aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

Site customer isUnknown

```
function enrichSite(aSite) {
  const result = _.cloneDeep(aSite);
  const unknownCustomer = {
    isUnknown: true,
  };

  if (isUnknown(result.customer)) result.customer = unknownCustomer;
  else result.customer.isUnknown = false;
  return result;
}
```

Site Site

```
function isUnknown(aCustomer) {
  if (aCustomer === "unknown") return true;
  else return aCustomer.isUnknown;
}
```

149 “ ”

```
function enrichSite(aSite) {
  const result = _.cloneDeep(aSite);
  const unknownCustomer = {
    isUnknown: true,
    name: "occupant",
  };

  if (isUnknown(result.customer)) result.customer = unknownCustomer;
  else result.customer.isUnknown = false;
  return result;
}
```

### 1...

```
const rawSite = acquireSiteData();
const site = enrichSite(rawSite);
const aCustomer = site.customer;
// ... lots of intervening code ...
const customerName = aCustomer.name;
```

“ ”

```
function enrichSite(aSite) {
  const result = _.cloneDeep(aSite);
  const unknownCustomer = {
    isUnknown: true,
    name: "occupant",
    billingPlan: registry.billingPlans.basic,
  };

  if (isUnknown(result.customer)) result.customer = unknownCustomer;
  else result.customer.isUnknown = false;
  return result;
}
```

## 2...

```
const plan = aCustomer.billingPlan;
```

```
function enrichSite(aSite) {
  const result = _.cloneDeep(aSite);
  const unknownCustomer = {
    isUnknown: true,
    name: "occupant",
    billingPlan: registry.billingPlans.basic,
    paymentHistory: {
      weeksDelinquentInLastYear: 0,
    },
  };

  if (isUnknown(result.customer)) result.customer = unknownCustomer;
  else result.customer.isUnknown = false;
  return result;
}
```

## 3...

```
const weeksDelinquent = aCustomer.paymentHistory.weeksDelinquentInLastYear;
```

# 10.6 Introduce Assertion

```
  if (this.discountRate)
  base = base - (this.discountRate * base);


  assert(this.discountRate>= 0);
if (this.discountRate)
  base = base - (this.discountRate * base);
```

null

——

——

“ ”

customer discount rate

## class Customer...

```
applyDiscount(aNumber) {
  return (this.discountRate)
    ? aNumber - (this.discountRate * aNumber)
    : aNumber;
}
```

if-else

## class Customer...

```
applyDiscount(aNumber) {
  if (!this.discountRate) return aNumber;
  else return aNumber - (this.discountRate * aNumber);
}
```

## class Customer...

```
applyDiscount(aNumber) {
  if (!this.discountRate) return aNumber;
  else {
    assert(this.discountRate >= 0);
    return aNumber - (this.discountRate * aNumber);
  }
}
```

applyDiscount

## class Customer...

```
set discountRate(aNumber) {
  assert(null === aNumber || aNumber >= 0);
  this._discountRate = aNumber;
}
```

——

“ ” “ ”

106

—— bug

# 11 API

API API API API

API 306
310 314

319
324 327

331
334

337 106
344

## 11.1 Separate Query from Modifier

```
function getTotalOutstandingAndSendBill() {
const result = customer.invoices.reduce((total, each) => each.amount + total, 0
sendBill();
return result;
}


function totalOutstanding() {
return customer.invoices.reduce((total, each) => each.amount + total, 0);

}
function sendBill() {
emailGateway.send(formatBill(customer));
}
```

“ ” “ ” —— Command-Query Separation [mf-cqs]

“ ”

“ ”

miscreant people

```
function alertForMiscreant(people) {
  for (const p of people) {
    if (p === "Don") {
      setOffAlarms();
      return "Don";
    }
    if (p === "John") {
      setOffAlarms();
      return "John";
    }
  }
  return "";
}
```

```
function findMiscreant(people) {
  for (const p of people) {
    if (p === "Don") {
      setOffAlarms();
      return "Don";
    }
    if (p === "John") {
      setOffAlarms();
      return "John";
    }
  }
  return "";
}
```

```
function findMiscreant(people) {
  for (const p of people) {
    if (p === "Don") {
      setOffAlarms();
      return "Don";
    }
    if (p === "John") {
      setOffAlarms();
      return "John";
    }
  }
  return "";
}
```

```
const found = alertForMiscreant(people);
```

```
const found = findMiscreant(people);
alertForMiscreant(people);
```

```
function alertForMiscreant(people) {
  for (const p of people) {
    if (p === "Don") {
      setOffAlarms();
      return;
    }
    if (p === "John") {
      setOffAlarms();
      return;
    }
  }
  return;
}
```



```
function alertForMiscreant(people) {
  if (findMiscreant(people) !== "") setOffAlarms();
}
```

## 11.2 Parameterize Function

Parameterize Method

```
function tenPercentRaise(aPerson) {
  aPerson.salary = aPerson.salary.multiply(1.1);
}
function fivePercentRaise(aPerson) {
  aPerson.salary = aPerson.salary.multiply(1.05);
}

function raise(aPerson, factor) {
  aPerson.salary = aPerson.salary.multiply(1 + factor);
}
```

124

```
function tenPercentRaise(aPerson) {
  aPerson.salary = aPerson.salary.multiply(1.1);
}
function fivePercentRaise(aPerson) {
  aPerson.salary = aPerson.salary.multiply(1.05);
}
```

```
function raise(aPerson, factor) {
  aPerson.salary = aPerson.salary.multiply(1 + factor);
}
```

```
function baseCharge(usage) {
 if (usage < 0) return usd(0);
 const amount =
    bottomBand(usage) * 0.03
    + middleBand(usage) * 0.05
    + topBand(usage) * 0.07;
 return usd(amount);
}

function bottomBand(usage) {
 return Math.min(usage, 100);
}

function middleBand(usage) {
 return usage > 100 ? Math.min(usage, 200) - 100 : 0;
}

function topBand(usage) {
 return usage > 200 ? usage - 200 : 0;
}
```

“ ” band

“ ”

middleBand

middleBand 100 200 “ ” 124

```
function withinBand(usage, bottom, top) {
 return usage > 100 ? Math.min(usage, 200) - 100 : 0;
}

function baseCharge(usage) {
 if (usage < 0) return usd(0);
 const amount =
    bottomBand(usage) * 0.03
    + withinBand(usage, 100, 200) * 0.05
    + topBand(usage) * 0.07;
 return usd(amount);
}
```

```
function withinBand(usage, bottom, top) {
  return usage & gt;
  bottom ? Math.min(usage, 200) - bottom : 0;
}
```

```
function withinBand(usage, bottom, top) {
  return usage & gt;
  bottom ? Math.min(usage, top) - bottom : 0;
}
```

bottomBand

```
function baseCharge(usage) {
 if (usage < 0) return usd(0);
 const amount =
    withinBand(usage, 0, 100) * 0.03
    + withinBand(usage, 100, 200) * 0.05
    + topBand(usage) * 0.07;
 return usd(amount);
}

function bottomBand(usage) {
 return Math.min(usage, 100);
}
```

topBand “ ” Infinity

```
function baseCharge(usage) {
 if (usage < 0) return usd(0);
 const amount =
    withinBand(usage, 0, 100) * 0.03
    + withinBand(usage, 100, 200) * 0.05
    + withinBand(usage, 200, Infinity) * 0.07;
 return usd(amount);
}

function topBand(usage) {
  return usage > 200 ? usage - 200 : 0;
}
```

baseCharge “ usage ”

## 11.3 Remove Flag Argument

Replace Parameter with Explicit Methods

```
function setDimension(name, value) {
  if (name === "height") {
    this._height = value;
    return;
  }
  if (name === "width") {
    this._width = value;
    return;
  }
}

function setHeight(value) {
  this._height = value;
}
function setWidth(value) {
  this._width = value;
}
```

“ ”

```
function bookConcert(aCustomer, isPremium) {
  if (isPremium) {
    // logic for premium booking
  } else {
    // logic for regular booking
  }
}
```

premium concert

```
bookConcert(aCustomer, true);
```

```
bookConcert(aCustomer, CustomerType.PREMIUM);
```

```
bookConcert(aCustomer, "premium");
```

API

—— true

```
premiumBookConcert(aCustomer);
```

“ ” “ ”

260

“ ”

shipment delivery date

```
aShipment.deliveryDate = deliveryDate(anOrder, true);
```

```
aShipment.deliveryDate = deliveryDate(anOrder, false);
```

deliveryDate

```
function deliveryDate(anOrder, isRush) {
  if (isRush) {
    let deliveryTime;
    if (["MA", "CT"].includes(anOrder.deliveryState)) deliveryTime = 1;
    else if (["NY", "NH"].includes(anOrder.deliveryState)) deliveryTime = 2;
    else deliveryTime = 3;
    return anOrder.placedOn.plusDays(1 + deliveryTime);
  } else {
    let deliveryTime;
    if (["MA", "CT", "NY"].includes(anOrder.deliveryState)) deliveryTime = 2;
    else if (["ME", "NH"].includes(anOrder.deliveryState)) deliveryTime = 3;
    else deliveryTime = 4;
    return anOrder.placedOn.plusDays(2 + deliveryTime);
  }
}
```

——

260

```
function deliveryDate(anOrder, isRush) {
  if (isRush) return rushDeliveryDate(anOrder);
  else return regularDeliveryDate(anOrder);
}
function rushDeliveryDate(anOrder) {
  let deliveryTime;
  if (["MA", "CT"].includes(anOrder.deliveryState)) deliveryTime = 1;
  else if (["NY", "NH"].includes(anOrder.deliveryState)) deliveryTime = 2;
  else deliveryTime = 3;
  return anOrder.placedOn.plusDays(1 + deliveryTime);
}
function regularDeliveryDate(anOrder) {
  let deliveryTime;
  if (["MA", "CT", "NY"].includes(anOrder.deliveryState)) deliveryTime = 2;
  else if (["ME", "NH"].includes(anOrder.deliveryState)) deliveryTime = 3;
  else deliveryTime = 4;
  return anOrder.placedOn.plusDays(2 + deliveryTime);
}
```

```
aShipment.deliveryDate = deliveryDate(anOrder, true);
```

```
aShipment.deliveryDate = rushDeliveryDate(anOrder);
```

deliveryDate

deliveryDate

```
const isRush = determineIfRush(anOrder);
aShipment.deliveryDate = deliveryDate(anOrder, isRush);
```

260

314

deliveryDate

“ ”

deliveryDate

```
function deliveryDate(anOrder, isRush) {
 let result;
 let deliveryTime;
 if (anOrder.deliveryState === "MA" || anOrder.deliveryState === "CT")
  deliveryTime = isRush? 1 : 2;
 else if (anOrder.deliveryState === "NY" || anOrder.deliveryState === "NH") {
  deliveryTime = 2;
  if (anOrder.deliveryState === "NH" &amp;amp;&amp;amp; !isRush)
   deliveryTime = 3;
 }
 else if (isRush)
  deliveryTime = 3;
 else if (anOrder.deliveryState === "ME")
  deliveryTime = 3;
 else
  deliveryTime = 4;
 result = anOrder.placedOn.plusDays(2 + deliveryTime);
 if (isRush) result = result.minusDays(1);
 return result;
}
```

isRush deliveryDate

```
function rushDeliveryDate(anOrder) {
  return deliveryDate(anOrder, true);
}
function regularDeliveryDate(anOrder) {
  return deliveryDate(anOrder, false);
}
```

deliveryDate

isRush

deliveryDateHelperOnly

## 11.4 Preserve Whole Object

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
if (aPlan.withinRange(low, high))


if (aPlan.withinRange(aRoom.daysTempRange))
```

“ ”

140

182

JavaScript this

“ ” 237

115

heating plan

...

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
if (!aPlan.withinRange(low, high))
  alerts.push("room temperature went outside range");
```

## class HeatingPlan...

```
withinRange(bottom, top) {
 return (bottom >= this._temperatureRange.low) &amp;&amp; (top <= this._tempera
}
```

“ ” withinRange

HeatingPlan

## class HeatingPlan...

```
xxNEWwithinRange(aNumberRange) {
}
```

withinRange

withinRange

## class HeatingPlan...

```
xxNEWwithinRange(aNumberRange) {
  return this.withinRange(aNumberRange.low, aNumberRange.high);
}
```

...

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
if (!aPlan.xxNEWwithinRange(aRoom.daysTempRange))
  alerts.push("room temperature went outside range");
```

237

...

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
if (!aPlan.xxNEWwithinRange(aRoom.daysTempRange))
  alerts.push("room temperature went outside range");
```

115

## class HeatingPlan...

```
xxNEWwithinRange(aNumberRange) {
 return (aNumberRange.low >= this._temperatureRange.low) &amp;&amp;
  (aNumberRange.high <= this._temperatureRange.high);
}
```

## class HeatingPlan...

```
withinRange(aNumberRange) {
 return (aNumberRange.low >= this._temperatureRange.low) &amp;&amp;
  (aNumberRange.high <= this._temperatureRange.high);
}
```

•••

```
if (!aPlan.withinRange(aRoom.daysTempRange))
  alerts.push("room temperature went outside range");
```

•••

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
if (!aPlan.withinRange(low, high))
  alerts.push("room temperature went outside range");
```

106 119

•••

```
const low = aRoom.daysTempRange.low;
const high = aRoom.daysTempRange.high;
const isWithinRange = aPlan.withinRange(low, high);
if (!isWithinRange) alerts.push("room temperature went outside range");
```

•••

```
const tempRange = aRoom.daysTempRange;
const low = tempRange.low;
const high = tempRange.high;
const isWithinRange = aPlan.withinRange(low, high);
if (!isWithinRange) alerts.push("room temperature went outside range");
```

106

•••

```
const tempRange = aRoom.daysTempRange;
const isWithinRange = xxNEWwithinRange(aPlan, tempRange);
if (!isWithinRange) alerts.push("room temperature went outside range");
```

...

```
function xxNEWwithinRange(aPlan, tempRange) {
  const low = tempRange.low;
  const high = tempRange.high;
  const isWithinRange = aPlan.withinRange(low, high);
  return isWithinRange;
}
```

HeatingPlan 198

...

```
const tempRange = aRoom.daysTempRange;
const isWithinRange = aPlan.xxNEWwithinRange(tempRange);
if (!isWithinRange) alerts.push("room temperature went outside range");
```

### class HeatingPlan...

```
xxNEWwithinRange(tempRange) {
  const low = tempRange.low;
  const high = tempRange.high;
  const isWithinRange = this.withinRange(low, high);
  return isWithinRange;
}
```

## 11.5 Replace Parameter with Query

Replace Parameter with Method

327

```
availableVacation(anEmployee, anEmployee.grade);

function availableVacation(anEmployee, grade) {
  // calculate vacation...


  availableVacation(anEmployee)

function availableVacation(anEmployee) {
  const grade = anEmployee.grade;
  // calculate vacation...
```

“ ” “ ”

____ “ ”

referential transparency

106

124

## class Order...

```
get finalPrice() {
 const basePrice = this.quantity * this.itemPrice;
 let discountLevel;
 if (this.quantity > 100) discountLevel = 2;
 else discountLevel = 1;
 return this.discountedPrice(basePrice, discountLevel);
}

discountedPrice(basePrice, discountLevel) {
 switch (discountLevel) {
  case 1: return basePrice * 0.95;
  case 2: return basePrice * 0.9;
 }
}
```

178

## class Order...

```
get finalPrice() {
 const basePrice = this.quantity * this.itemPrice;
 return this.discountedPrice(basePrice, this.discountLevel);
}

get discountLevel() {
 return (this.quantity > 100) ? 2 : 1;
}
```

discountLevel discountedPrice discountLevel

discountedPrice discountLevel

## class Order...

```
discountedPrice(basePrice, discountLevel) {
 switch (this.discountLevel) {
  case 1: return basePrice * 0.95;
  case 2: return basePrice * 0.9;
 }
}
```

124

## class Order...

```
get finalPrice() {
 const basePrice = this.quantity * this.itemPrice;
 return this.discountedPrice(basePrice, this.discountLevel);
}

discountedPrice(basePrice, discountLevel) {
 switch (this.discountLevel) {
  case 1: return basePrice * 0.95;
  case 2: return basePrice * 0.9;
 }
}
```

## 11.6 Replace Query with Parameter

324

```
  targetTemperature(aPlan)

function targetTemperature(aPlan) {
  currentTemperature = thermostat.currentTemperature;
  // rest of function...


  targetTemperature(aPlan, thermostat.currentTemperature)

function targetTemperature(aPlan, currentTemperature) {
  // rest of function...
```

——

“ ”

“ ” referential transparency

“ ”

I/O

——

119

106

123

115

thermostat heating plan

## class HeatingPlan...

```
get targetTemperature() {
  if (thermostat.selectedTemperature > this._max) return this._max;
  else if (thermostat.selectedTemperature < this._min) return this._min;
  else return thermostat.selectedTemperature;
}
```

**...**

```
if (thePlan.targetTemperature > thermostat.currentTemperature) setToHeat();
else if (thePlan.targetTemperature<thermostat.currentTemperature)setToCool();
else setOff();
```

targetTemperature thermostat

119 “ ”

## class HeatingPlan...

```
get targetTemperature() {
 const selectedTemperature = thermostat.selectedTemperature;
 if      (selectedTemperature > this._max) return this._max;
 else if (selectedTemperature < this._min) return this._min;
 else return selectedTemperature;
}
```

106 “ ”

## class HeatingPlan...

```
get targetTemperature() {
 const selectedTemperature = thermostat.selectedTemperature;
 return this.xxNEWtargetTemperature(selectedTemperature);
}

xxNEWtargetTemperature(selectedTemperature) {
 if      (selectedTemperature > this._max) return this._max;
 else if (selectedTemperature < this._min) return this._min;
 else return selectedTemperature;
}
```

## class HeatingPlan...

```
get targetTemperature() {
  return this.xxNEWtargetTemperature(thermostat.selectedTemperature);
}
```

115

•••

```
if (thePlan.xxNEWtargetTemperature(thermostat.selectedTemperature) >
   thermostat.currentTemperature)
 setToHeat();
else if (thePlan.xxNEWtargetTemperature(thermostat.selectedTemperature) <
     thermostat.currentTemperature)
 setToCool();
else
 setOff();
```

•••

```
if (thePlan.targetTemperature(thermostat.selectedTemperature) >
   thermostat.currentTemperature)
 setToHeat();
else if (thePlan.targetTemperature(thermostat.selectedTemperature) <
     thermostat.currentTemperature)
 setToCool();
else
 setOff();
```

## class HeatingPlan...

```
targetTemperature(selectedTemperature) {
 if (selectedTemperature > this._max) return this._max;
 else if (selectedTemperature < this._min) return this._min;
 else return selectedTemperature;
}
```

thermostat HeatingPlan ——
HeatingPlan thermostat targetTemperature
HeatingPlan targetTemperature HeatingPlan

JavaScript ——

## 11.7 Remove Setting Method

```
class Person {
get name() {...}
set name(aString) {...}


class Person {
get name() {...}
```

“ ” ——

“
”

“ ”

124

“ ” “ ”

115

Person

## class Person...

```
get name() {return this._name;}
set name(arg) {this._name = arg;}
get id() {return this._id;}
set id(arg) {this._id = arg;}
```

```
const martin = new Person();
martin.name = "martin";
martin.id = "1234";
```

name id id

id 124

## class Person...

```
constructor(id) {
  this.id = id;
}
```

id

```
const martin = new Person("1234");
martin.name = "martin";
martin.id = "1234";
```

Person

115

## class Person...

```
constructor(id) {
  this._id = id;
}
get name() {return this._name;}
set name(arg) {this._name = arg;}
get id() {return this._id;}
set id(arg) {this._id = arg;}
```

# 11.8 Replace Constructor with Factory Function

Replace Constructor with Factory Method

```
leadEngineer = new Employee(document.leadEngineer, "E");

leadEngineer = createEngineer(document.leadEngineer);
```

Java

new

Employee “ ”

## class Employee...

```
constructor (name, typeCode) {
  this._name = name;
  this._typeCode = typeCode;
}
get name() {return this._name;}
get type() {
  return Employee.legalTypeCodes[this._typeCode];
}
static get legalTypeCodes() {
  return {"E": "Engineer", "M": "Manager", "S": "Salesman"};
}
```

...

```
candidate = new Employee(document.name, document.empType);
```

•••

```
const leadEngineer = new Employee(document.leadEngineer, "E");
```

•••

```
function createEmployee(name, typeCode) {
  return new Employee(name, typeCode);
}
```

•••

```
candidate = createEmployee(document.name, document.empType);
```

•••

```
const leadEngineer = createEmployee(document.leadEngineer, "E");
```

____ “ ”

•••

```
const leadEngineer = createEngineer(document.leadEngineer);
```

•••

```
function createEngineer(name) {
  return new Employee(name, "E");
}
```

## 11.9 Replace Function with Command

Replace Method with Method Object

344

```
function score(candidate, medicalExam, scoringGuide) {
  let result = 0;
  let healthLevel = 0;
  // long body code
}

class Scorer {
  constructor(candidate, medicalExam, scoringGuide) {
    this._candidate = candidate;
    this._medicalExam = medicalExam;
    this._scoringGuide = scoringGuide;
  }

  execute() {
    this._result = 0;
    this._healthLevel = 0;
    // long body code
  }
}
```

method “ ”
command object “ ” command

95%

“ ” “ ” [gof] command
pattern “ ” “ ” “ ” command-query
separation principle “ ” “ ” “ ”
“ ” modifier “ ” mutator

198

“execute” “call”

JavaScript JavaScript
JavaScript

```
function score(candidate, medicalExam, scoringGuide) {
  let result = 0;
  let healthLevel = 0;
  let highMedicalRiskFlag = false;

  if (medicalExam.isSmoker) {
    healthLevel += 10;
    highMedicalRiskFlag = true;
  }
  let certificationGrade = "regular";
  if (scoringGuide.stateWithLowCertification(candidate.originState)) {
    certificationGrade = "low";
    result -= 5;
  } // lots more code like this
  result -= Math.max(healthLevel - 5, 0);
  return result;
}
```

198

```
function score(candidate, medicalExam, scoringGuide) {
  return new Scorer().execute(candidate, medicalExam, scoringGuide);
}

class Scorer {
  execute(candidate, medicalExam, scoringGuide) {
    let result = 0;
    let healthLevel = 0;
    let highMedicalRiskFlag = false;

    if (medicalExam.isSmoker) {
      healthLevel += 10;
      highMedicalRiskFlag = true;
    }
    let certificationGrade = "regular";
    if (scoringGuide.stateWithLowCertification(candidate.originState)) {
      certificationGrade = "low";
      result -= 5;
    } // lots more code like this
    result -= Math.max(healthLevel - 5, 0);
    return result;
  }
}
```

execute

execute

```
function score(candidate, medicalExam, scoringGuide) {
  return new Scorer(candidate).execute(candidate, medicalExam, scoringGuide);
}
```

## class Scorer...

```
constructor(candidate){
 this._candidate = candidate;
}

execute (candidate, medicalExam, scoringGuide) {
 let result = 0;
 let healthLevel = 0;
 let highMedicalRiskFlag = false;

 if (medicalExam.isSmoker) {
  healthLevel += 10;
  highMedicalRiskFlag = true;
 }
 let certificationGrade = "regular";
 if (scoringGuide.stateWithLowCertification(this._candidate.originState)) {
  certificationGrade = "low";
  result -= 5;
 }
 // lots more code like this
 result -= Math.max(healthLevel - 5, 0);
 return result;
}
```

```
function score(candidate, medicalExam, scoringGuide) {
  return new Scorer(candidate, medicalExam, scoringGuide).execute();
}
```

## class Scorer...

```
constructor(candidate, medicalExam, scoringGuide){
 this._candidate = candidate;
 this._medicalExam = medicalExam;
 this._scoringGuide = scoringGuide;
}
execute () {
 let result = 0;
 let healthLevel = 0;
 let highMedicalRiskFlag = false;

 if (this._medicalExam.isSmoker) {
  healthLevel += 10;
  highMedicalRiskFlag = true;
 }
 let certificationGrade = "regular";
 if (this._scoringGuide.stateWithLowCertification(this._candidate.originState)
  certificationGrade = "low";
  result -= 5;
 }
 // lots more code like this
 result -= Math.max(healthLevel - 5, 0);
 return result;
}
```

## class Scorer...

```
constructor(candidate, medicalExam, scoringGuide){
 this._candidate = candidate;
 this._medicalExam = medicalExam;
 this._scoringGuide = scoringGuide;
}

execute () {
 this._result = 0;
 let healthLevel = 0;
 let highMedicalRiskFlag = false;

 if (this._medicalExam.isSmoker) {
  healthLevel += 10;
  highMedicalRiskFlag = true;
 }
 let certificationGrade = "regular";
 if (this._scoringGuide.stateWithLowCertification(this._candidate.originState)
  certificationGrade = "low";
  this._result -= 5;
 }
 // lots more code like this
 this._result -= Math.max(healthLevel - 5, 0);
 return this._result;
}
```

“ ”

## class Scorer...

```
constructor(candidate, medicalExam, scoringGuide){
 this._candidate = candidate;
 this._medicalExam = medicalExam;
 this._scoringGuide = scoringGuide;
}

execute () {
 this._result = 0;
 this._healthLevel = 0;
 this._highMedicalRiskFlag = false;

 if (this._medicalExam.isSmoker) {
  this._healthLevel += 10;
  this._highMedicalRiskFlag = true;
 }
 this._certificationGrade = "regular";
 if (this._scoringGuide.stateWithLowCertification(this._candidate.originState))
  this._certificationGrade = "low";
  this._result -= 5;
 }
 // lots more code like this
 this._result -= Math.max(this._healthLevel - 5, 0);
 return this._result;
}
```

106

## class Scorer...

```
execute () {
 this._result = 0;
 this._healthLevel = 0;
 this._highMedicalRiskFlag = false;

 this.scoreSmoking();
 this._certificationGrade = "regular";
 if (this._scoringGuide.stateWithLowCertification(this._candidate.originState))
  this._certificationGrade = "low";
  this._result -= 5;
 }
 // lots more code like this
 this._result -= Math.max(this._healthLevel - 5, 0);
 return this._result;
 }
scoreSmoking() {
 if (this._medicalExam.isSmoker) {
  this._healthLevel += 10;
  this._highMedicalRiskFlag = true;
 }
}
```

JavaScript

# 11.10 Replace Command with Function

337

```
class ChargeCalculator {
  constructor(customer, usage) {
    this._customer = customer;
    this._usage = usage;
  }
  execute() {
    return this._customer.rate * this._usage;
  }
}

function charge(customer, usage) {
  return customer.rate * usage;
}
```

106 “ ”

115

119 115

124

“ ” “ ”

237

```
class ChargeCalculator {
  constructor(customer, usage, provider) {
    this._customer = customer;
    this._usage = usage;
    this._provider = provider;
  }
  get baseCharge() {
    return this._customer.baseRate * this._usage;
  }
  get charge() {
    return this.baseCharge + this._provider.connectionCharge;
  }
}
```

•••

```
monthCharge = new ChargeCalculator(customer, usage, provider).charge;
```

106

•••

```
monthCharge = charge(customer, usage, provider);
```

...

```
function charge(customer, usage, provider) {
  return new ChargeCalculator(customer, usage, provider).charge;
}
```

baseCharge 119

## class ChargeCalculator...

```
get baseCharge() {
  return this._customer.baseRate * this._usage;
}
get charge() {
  const baseCharge = this.baseCharge;
  return baseCharge + this._provider.connectionCharge;
}
```

115

## class ChargeCalculator...

```
get charge() {
  const baseCharge = this._customer.baseRate * this._usage;
  return baseCharge + this._provider.connectionCharge;
}
```

124 charge

## class ChargeCalculator...

```
constructor (customer, usage, provider){
 this._customer = customer;
 this._usage = usage;
 this._provider = provider;
}

charge(customer, usage, provider) {
 const baseCharge = this._customer.baseRate * this._usage;
 return baseCharge + this._provider.connectionCharge;
}
```

...

```
function charge(customer, usage, provider) {
  return new ChargeCalculator(customer, usage, provider).charge(
    customer,
    usage,
    provider
  );
}
```

charge

## class ChargeCalculator...

```
constructor (customer, usage, provider){
 this._customer = customer;
 this._usage = usage;
 this._provider = provider;
}

charge(customer, usage, provider) {
 const baseCharge = customer.baseRate * this._usage;
 return baseCharge + this._provider.connectionCharge;
}
```

`this._customer`

charge

## class ChargeCalculator...

```
charge(customer, usage, provider) {
  const baseCharge = customer.baseRate * usage;
  return baseCharge + provider.connectionCharge;
}
```

charge 115

•••

```
function charge(customer, usage, provider) {
  const baseCharge = customer.baseRate * usage;
  return baseCharge + provider.connectionCharge;
}
```

237

# 12

350 353 355 359
361 375 369 380
362

—— 381 399

## 12.1 Pull Up Method

359

```
class Employee {...}

class Salesman extends Employee {
 get name() {...}
}

class Engineer extends Employee {
 get name() {...}
}


class Employee {
 get name() {...}
}

class Salesman extends Employee {...}
class Engineer extends Employee {...}
```

bug “ ”

310

353

Form Template Method [mf-ft]

124

### class Employee extends Party...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

### class Department extends Party...

```
get totalAnnualCost() {
  return this.monthlyCost * 12;
}
```

monthlyCost JavaScript
monthlyCost Party

124

### class Department...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

annualCost

### class Party...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

JavaScript Employee annualCost Department annualCost

annualCost monthlyCost Party JavaScript “ Party monthlyCost ” trap

### class Party...

```
get monthlyCost() {
  throw new SubclassResponsibilityError();
}
```

“ ” Smalltalk

## 12.2 Pull Up Field

361

```
class Employee {...} // Java

class Salesman extends Employee {
 private String name;
}

class Engineer extends Employee {
 private String name;
}


class Employee {
 protected String name;
}

class Salesman extends Employee {...}
class Engineer extends Employee {...}
```

137

protected

## 12.3 Pull Up Constructor Body

```
class Party {...}

class Employee extends Party {
 constructor(name, id, monthlyCost) {
  super();
  this._id = id;
  this._name = name;
  this._monthlyCost = monthlyCost;
 }
}


class Party {
 constructor(name){
  this._name = name;
 }
}

class Employee extends Party {
 constructor(name, id, monthlyCost) {
  super(name);
  this._id = id;
  this._monthlyCost = monthlyCost;
 }
}
```

106 350

334

223

106 350

“ ”

```
class Party {}

class Employee extends Party {
 constructor(name, id, monthlyCost) {
  super();
  this._id = id;
  this._name = name;
  this._monthlyCost = monthlyCost;
 }
 // rest of class...
class Department extends Party {
 constructor(name, staff){
  super();
  this._name = name;
  this._staff = staff;
 }
 // rest of class...
```

Party name 223 Employee super()

```
class Employee extends Party {
  constructor(name, id, monthlyCost) {
    super();
    this._name = name;
    this._id = id;
    this._monthlyCost = monthlyCost;
  }
  // rest of class...
```

name

## class Party...

```
constructor(name){
  this._name = name;
}
```

## class Employee...

```
constructor(name, id, monthlyCost) {
  super(name);
  this._id = id;
  this._monthlyCost = monthlyCost;
}
```

## class Department...

```
constructor(name, staff){
  super(name);
  this._staff = staff;
}
```

super

## class Employee...

```
constructor (name) {...}

get isPrivileged() {...}

assignCar() {...}
```

## class Manager extends Employee...

```
constructor(name, grade) {
  super(name);
  this._grade = grade;
  if (this.isPrivileged) this.assignCar(); // every subclass does this
}

get isPrivileged() {
  return this._grade >4;
}
```

isPrivileged grade

106

## class Manager...

```
constructor(name, grade) {
  super(name);
  this._grade = grade;
  this.finishConstruction();
}

finishConstruction() {
  if (this.isPrivileged) this.assignCar();
}
```

350

### class Employee...

```
finishConstruction() {
  if (this.isPrivileged) this.assignCar();
}
```

## 12.4 Push Down Method

350

```
class Employee {
  get quota {...}
}

class Engineer extends Employee {...}
class Salesman extends Employee {...}


class Employee {...}
class Engineer extends Employee {...}
class Salesman extends Employee {
  get quota {...}
}
```

272

## 12.5 Push Down Field

353

```
class Employee {   // Java
 private String quota;
}

class Engineer extends Employee {...}
class Salesman extends Employee {...}


class Employee {...}
class Engineer extends Employee {...}

class Salesman extends Employee {
 protected String quota;
}
```

## 12.6 Replace Type Code with Subclasses

State/Strategy Replace Type Code with State/Strategy

Extract Subclass

369

```
function createEmployee(name, type) {
  return new Employee(name, type);
}


function createEmployee(name, type) {
  switch (type) {
    case "engineer": return new Engineer(name);
    case "salesman": return new Salesman(name);
    case "manager": return new Manager (name);
}
```

“ ” ——

272

“ ” “ ” 361

“ ” “ ” “ ” “ ” “ ” “ ” “ ” “ ” 174 “ ” 362

334

“ ”

359 272

……

**class Employee...**

```
constructor(name, type){
  this.validateType(type);
  this._name = name;
  this._type = type;
}
validateType(arg) {
  if (!["engineer", "manager", "salesman"].includes(arg))
    throw new Error(`Employee cannot be of type ${arg}`);
}
toString() {return `${this._name} (${this._type})`;}
```

132

## class Employee...

```
get type() {return this._type;}
toString() {return `${this._name} (${this.type})`;}
```

toString `this._type`

"engineer" Employee

```
class Engineer extends Employee {
  get type() {
    return "engineer";
  }
}
```

JavaScript 334

```
function createEmployee(name, type) {
  return new Employee(name, type);
}
```

```
function createEmployee(name, type) {
  switch (type) {
    case "engineer":
      return new Engineer(name, type);
  }
  return new Employee(name, type);
}
```

Engineer type

type

```
class Salesman extends Employee {
  get type() {
    return "salesman";
  }
}

class Manager extends Employee {
  get type() {
    return "manager";
  }
}

function createEmployee(name, type) {
  switch (type) {
    case "engineer":
      return new Engineer(name, type);
    case "salesman":
      return new Salesman(name, type);
    case "manager":
      return new Manager(name, type);
  }
  return new Employee(name, type);
}
```

## class Employee...

```
constructor(name, type){
 this.validateType(type);
 this._name = name;
 this._type = type;
}

get type() {return this._type;}
toString() {return `${this._name} (${this.type})`;}
```

## class Employee...

```
constructor(name, type){
 this.validateType(type);
 this._name = name;
}
function createEmployee(name, type) {
 switch (type) {
  case "engineer": return new Engineer(name, type);
  case "salesman": return new Salesman(name, type);
  case "manager":  return new Manager (name, type);
  default: throw new Error(`Employee cannot be of type ${type}`);
 }
 return new Employee(name, type);
}
```

124

## class Employee...

```
constructor(name, type){
 this._name = name;
}

function createEmployee(name, type) {
 switch (type) {
  case "engineer": return new Engineer(name, type);
  case "salesman": return new Salesman(name, type);
  case "manager": return new Manager (name, type);
  default: throw new Error(`Employee cannot be of type ${type}`);
 }
}
```

——get type ——

272 359 237

“ ” “ ”

## class Employee...

```
constructor(name, type){
 this.validateType(type);
 this._name = name;
 this._type = type;
}
validateType(arg) {
 if (!["engineer", "manager", "salesman"].includes(arg))
  throw new Error(`Employee cannot be of type ${arg}`);
}
get type()    {return this._type;}
set type(arg) {this._type = arg;}

get capitalizedType() {
 return this._type.charAt(0).toUpperCase() + this._type.substr(1).toLowerCase()
}
toString() {
 return `${this._name} (${this.capitalizedType})`;
}
```

toString

174

```
class EmployeeType {
  constructor(aString) {
    this._value = aString;
  }
  toString() {
    return this._value;
  }
}
```

## class Employee...

```
constructor(name, type){
 this.validateType(type);
 this._name = name;
 this.type = type;
}
validateType(arg) {
 if (!["engineer", "manager", "salesman"].includes(arg))
  throw new Error(`Employee cannot be of type ${arg}`);
}
get typeString()  {return this._type.toString();}
get type()    {return this._type;}
set type(arg) {this._type = new EmployeeType(arg);}

get capitalizedType() {
 return this.typeString.charAt(0).toUpperCase()
  + this.typeString.substr(1).toLowerCase();
}
toString() {
 return `${this._name} (${this.capitalizedType})`;
}
```

362

## class Employee...

```
set type(arg) {this._type = Employee.createEmployeeType(arg);}

 static createEmployeeType(aString) {
  switch(aString) {
   case "engineer": return new Engineer();
   case "manager":  return new Manager ();
   case "salesman": return new Salesman();
   default: throw new Error(`Employee cannot be of type ${aString}`);
  }
 }

class EmployeeType {
}
class Engineer extends EmployeeType {
 toString() {return "engineer";}
}
class Manager extends EmployeeType {
 toString() {return "manager";}
}
class Salesman extends EmployeeType {
 toString() {return "salesman";}
}
```

EmployeeType toString

“ ”

### class Employee...

```
toString() {
  return `${this._name} (${this.type.capitalizedName})`;
}
```

### class EmployeeType...

```
get capitalizedName() {
return this.toString().charAt(0).toUpperCase()
  + this.toString().substr(1).toLowerCase();
}
```

1 1 State/Strategy

## 12.7 Remove Subclass

Replace Subclass with Fields

362

```
class Person {
  get genderCode() {
    return "X";
  }
}
class Male extends Person {
  get genderCode() {
    return "M";
  }
}
class Female extends Person {
  get genderCode() {
    return "F";
  }
}

class Person {
  get genderCode() {
    return this._genderCode;
  }
}
```

334

106 198

**class Person...**

```
constructor(name) {
 this._name = name;
}
get name()    {return this._name;}
get genderCode() {return "X";}
// snip

class Male extends Person {
 get genderCode() {return "M";}
}

class Female extends Person {
 get genderCode() {return "F";}
}
```

**...**

```
const numberOfMales = people.filter(p => p instanceof Male).length;
```

“ ” 334

```
function createPerson(name) {
  return new Person(name);
}
function createMale(name) {
  return new Male(name);
}
function createFemale(name) {
  return new Female(name);
}
```

```
function loadFromInput(data) {
 const result = [];
 data.forEach(aRecord => {
  let p;
  switch (aRecord.gender) {
   case 'M': p = new Male(aRecord.name); break;
   case 'F': p = new Female(aRecord.name); break;
   default: p = new Person(aRecord.name);
  }
  result.push(p);
 });
 return result;
}
```

106 “ ”

```
function createPerson(aRecord) {
 let p;
 switch (aRecord.gender) {
  case 'M': p = new Male(aRecord.name); break;
  case 'F': p = new Female(aRecord.name); break;
  default: p = new Person(aRecord.name);
 }
 return p;
}
function loadFromInput(data) {
 const result = [];
 data.forEach(aRecord => {
  result.push(createPerson(aRecord));
 });
 return result;
}
```

123 createPerson

```
function createPerson(aRecord) {
  switch (aRecord.gender) {
    case "M":
      return new Male(aRecord.name);
    case "F":
      return new Female(aRecord.name);
    default:
      return new Person(aRecord.name);
  }
}
```

231 loadFromInput

```
function loadFromInput(data) {
  return data.map(aRecord => createPerson(aRecord));
}
```

instanceof —— 106

•••

```
const numberOfMales = people.filter(p => isMale(p)).length;

function isMale(aPerson) {return aPerson instanceof Male;}
```

198 Person

## class Person...

```
get isMale() {return this instanceof Male;}
```

•••

```
const numberOfMales = people.filter(p => p.isMale).length;
```

“ ”

## class Person...

```
constructor(name, genderCode) {
  this._name = name;
  this._genderCode = genderCode || "X";
}

get genderCode() {return this._genderCode;}
```

“ ” Person instanceof

```
function createPerson(aRecord) {
  switch (aRecord.gender) {
    case "M":
      return new Person(aRecord.name, "M");
    case "F":
      return new Female(aRecord.name);
    default:
      return new Person(aRecord.name);
  }
}
```

## class Person...

```
get isMale() {return "M" === this._genderCode;}
```

Male Female

```
function createPerson(aRecord) {
  switch (aRecord.gender) {
    case "M":
      return new Person(aRecord.name, "M");
    case "F":
      return new Person(aRecord.name, "F");
    default:
      return new Person(aRecord.name);
  }
}
```

```
function createPerson(aRecord) {
  switch (aRecord.gender) {
    case "M":
      return new Person(aRecord.name, "M");
    case "F":
      return new Person(aRecord.name, "F");
    default:
      return new Person(aRecord.name, "X");
  }
}
```

### class Person...

```
constructor(name, genderCode) {
  this._name = name;
  this._genderCode = genderCode || "X";
}
```

## 12.8 Extract Superclass

```
class Department {
 get totalAnnualCost() {...}
 get name() {...}
 get headCount() {...}
}

class Employee {
 get annualCost() {...}
 get name() {...}
 get id() {...}
}


class Party {
 get name() {...}
 get annualCost() {...}
}

class Department extends Party {
 get annualCost() {...}
 get headCount() {...}
}

class Employee extends Party {
 get annualCost() {...}
 get id() {...}
}
```

353 350

“ ”

182

399

124

355 350 353

106 350

—— name monthly cost annual cost

```
class Employee {
 constructor(name, id, monthlyCost) {
  this._id = id;
  this._name = name;
  this._monthlyCost = monthlyCost;
 }
 get monthlyCost() {return this._monthlyCost;}
 get name() {return this._name;}
 get id() {return this._id;}

 get annualCost() {
  return this.monthlyCost * 12;
 }
}

class Department {
 constructor(name, staff){
  this._name = name;
  this._staff = staff;
 }
 get staff() {return this._staff.slice();}
 get name() {return this._name;}

 get totalMonthlyCost() {
  return this.staff
    .map(e => e.monthlyCost)
    .reduce((sum, cost) => sum + cost);
 }
 get headCount() {
  return this.staff.length;
 }
 get totalAnnualCost() {
  return this.totalMonthlyCost * 12;
 }
}
```

```
class Party {}

class Employee extends Party {
 constructor(name, id, monthlyCost) {
  super();
  this._id = id;
  this._name = name;
  this._monthlyCost = monthlyCost;
 }
 // rest of class...
class Department extends Party {
 constructor(name, staff){
  super();
  this._name = name;
  this._staff = staff;
 }
 // rest of class...
```

JavaScript 353 name

## class Party...

```
constructor(name){
  this._name = name;
}
```

## class Employee...

```
constructor(name, id, monthlyCost) {
  super(name);
  this._id = id;
  this._monthlyCost = monthlyCost;
}
```

## class Department...

```
constructor(name, staff){
  super(name);
  this._staff = staff;
}
```

350 name

## class Party...

```
get name() {return this._name;}
```

## class Employee...

```
get name() {return this._name;}
```

## class Department...

```
get name() {return this._name;}
```

## class Employee...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

## class Department...

```
get totalAnnualCost() {
  return this.totalMonthlyCost * 12;
}
```

monthlyCost totalMonthlyCost 124

## class Department...

```
get totalAnnualCost() {
  return this.monthlyCost * 12;
}

get monthlyCost() { ... }
```

## class Department...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

350

## class Party...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

### class Employee...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

### class Department...

```
get annualCost() {
  return this.monthlyCost * 12;
}
```

## 12.9 Collapse Hierarchy

```
class Employee {...}
class Salesman extends Employee {...}


class Employee {...}
```

353 361 350 359

## 12.10 Replace Subclass with Delegate

```
class Order {
  get daysToShip() {
    return this._warehouse.daysToShip;
  }
}

class PriorityOrder extends Order {
  get daysToShip() {
    return this._priorityPlan.daysToShip;
  }
}
```

```
class Order {
  get daysToShip() {
    return this._priorityDelegate
      ? this._priorityDelegate.daysToShip
      : this._warehouse.daysToShip;
  }
}

class PriorityOrderDelegate {
  get daysToShip() {
    return this._priorityPlan.daysToShip;
  }
}
```

“ ” “ ” “ ”

“ ” “ ” “ ” “ ”

“ ” “ ” “ ” “ ” “ ”

——

[gof] “ ”

State Strategy

198

237

375

237

show booking

## class Booking...

```
constructor(show, date) {
  this._show = show;
  this._date = date;
}
```

premium extra

## class PremiumBooking extends Booking...

```
constructor(show, date, extras) {
  super(show, date);
  this._extras = extras;
}
```

PremiumBooking “ ” programming-by-difference

“ ” talkback

## class Booking...

```
get hasTalkback() {
 return this._show.hasOwnProperty('talkback') &amp;amp;&amp;amp; !this.isPeakDay;
}
```

PremiumBooking

## class PremiumBooking...

```
get hasTalkback() {
  return this._show.hasOwnProperty('talkback');
}
```

PremiumBooking

## class Booking...

```
get basePrice() {
  let result = this._show.price;
  if (this.isPeakDay) result += Math.round(result * 0.15);
  return result;
}
```

## class PremiumBooking...

```
get basePrice() {
  return Math.round(super.basePrice + this._extras.premiumFee);
}
```

PremiumBooking

## class PremiumBooking...

```
get hasDinner() {
  return this._extras.hasOwnProperty('dinner') &amp;amp;&amp;amp; !this.isPeakDay;
}
```

____

____

“ ” ____

“ ”

aBooking.bePremium() HTTP

“ ” Booking “ ” “

”

```
aBooking = new Booking(show, date);
```

```
aBooking = new PremiumBooking(show, date, extras);
```

334

•••

```
function createBooking(show, date) {
  return new Booking(show, date);
}
function createPremiumBooking(show, date, extras) {
  return new PremiumBooking(show, date, extras);
}
```

```
aBooking = createBooking(show, date);
```

```
aBooking = createPremiumBooking(show, date, extras);
```

Booking

## class PremiumBookingDelegate...

```
constructor(hostBooking, extras) {
  this._host = hostBooking;
  this._extras = extras;
}
```

Booking “ ”

•••

```
function createPremiumBooking(show, date, extras) {
  const result = new PremiumBooking(show, date, extras);
  result._bePremium(extras);
  return result;
}
```

## class Booking...

```
_bePremium(extras) {
  this._premiumDelegate = new PremiumBookingDelegate(this, extras);
}
```

`_bePremium` Booking

Booking “ ” extras

hasTalkback

## class Booking...

```
get hasTalkback() {
  return this._show.hasOwnProperty('talkback') &amp;amp; !this.isPeakDay;
}
```

## class PremiumBooking...

```
get hasTalkback() {
  return this._show.hasOwnProperty('talkback');
}
```

198 _host

## class PremiumBookingDelegate...

```
get hasTalkback() {
  return this._host._show.hasOwnProperty('talkback');
}
```

## class PremiumBooking...

```
get hasTalkback() {
  return this._premiumDelegate.hasTalkback;
}
```

## class PremiumBooking...

```
get hasTalkback() {
  return this._premiumDelegate.hasTalkback;
}
```

hasTalkback

## class Booking...

```
get hasTalkback() {
  return (this._premiumDelegate)
    ? this._premiumDelegate.hasTalkback
    : this._show.hasOwnProperty('talkback') &amp;amp;&amp;amp; !this.isPeakDay;
}
```

basePrice

## class Booking...

```
get basePrice() {
  let result = this._show.price;
  if (this.isPeakDay) result += Math.round(result * 0.15);
  return result;
}
```

## class PremiumBooking...

```
get basePrice() {
  return Math.round(super.basePrice + this._extras.premiumFee);
}
```

“ ” ____

`this._host.basePrice` `_host` PremiumBooking

106 “ ”

## class Booking...

```
get basePrice() {
 return (this._premiumDelegate)
  ? this._premiumDelegate.basePrice
  : this._privateBasePrice;
}

get _privateBasePrice() {
 let result = this._show.price;
 if (this.isPeakDay) result += Math.round(result * 0.15);
 return result;
}
```

## class PremiumBookingDelegate...

```
get basePrice() {
  return Math.round(this._host._privateBasePrice + this._extras.premiumFee);
}
```

## class Booking...

```
get basePrice() {
  let result = this._show.price;
  if (this.isPeakDay) result += Math.round(result * 0.15);
  return (this._premiumDelegate)
    ? this._premiumDelegate.extendBasePrice(result)
    : result;
}
```

## class PremiumBookingDelegate...

```
extendBasePrice(base) {
  return Math.round(base + this._extras.premiumFee);
}
```

## class PremiumBooking...

```
get hasDinner() {
  return this._extras.hasOwnProperty('dinner') &amp;amp;&amp;amp; !this.isPeakDay;
}
```

## class PremiumBookingDelegate...

```
get hasDinner() {
  return this._extras.hasOwnProperty('dinner') &amp;amp;&amp;amp; !this._host.isPeakDay
}
```

Booking

## class Booking...

```
get hasDinner() {
  return (this._premiumDelegate)
    ? this._premiumDelegate.hasDinner
    : undefined;
}
```

JavaScript undefined

•••

```
function createPremiumBooking(show, date, extras) {
  const result = new PremiumBooking(show, date, extras);
  result._bePremium(extras);
  return result;
}
```

## class PremiumBooking extends Booking ...

“ ”

```
function createBird(data) {
 switch (data.type) {
  case 'EuropeanSwallow':
   return new EuropeanSwallow(data);
  case 'AfricanSwallow':
   return new AfricanSwallow(data);
  case 'NorweigianBlueParrot':
   return new NorwegianBlueParrot(data);
  default:
   return new Bird(data);
 }
}

class Bird {
 constructor(data) {
  this._name = data.name;
  this._plumage = data.plumage;
 }
 get name()  {return this._name;}

 get plumage() {
  return this._plumage || "average";
 }
 get airSpeedVelocity() {return null;}
}

class EuropeanSwallow extends Bird {
 get airSpeedVelocity() {return 35;}
}

class AfricanSwallow extends Bird {
 constructor(data) {
  super (data);
  this._numberOfCoconuts = data.numberOfCoconuts;
 }
 get airSpeedVelocity() {
  return 40 - 2 * this._numberOfCoconuts;
 }
}

class NorwegianBlueParrot extends Bird {
 constructor(data) {
  super (data);
  this._voltage = data.voltage;
  this._isNailed = data.isNailed;
 }

 get plumage() {
  if (this._voltage > 100) return "scorched";
  else return this._plumage || "beautiful";
 }
```

```
 get airSpeedVelocity() {
  return (this._isNailed) ? 0 : 10 + this._voltage / 10;
 }
}
```

bird “ ” wild “ ” captive Bird
WildBird CaptiveBird “ ” “ ” “ ”

—— EuropeanSwallow

```
class EuropeanSwallowDelegate {}
```

data
data.type

## class Bird...

```
constructor(data) {
 this._name = data.name;
 this._plumage = data.plumage;
 this._speciesDelegate = this.selectSpeciesDelegate(data);
}

 selectSpeciesDelegate(data) {
  switch(data.type) {
   case 'EuropeanSwallow':
    return new EuropeanSwallowDelegate();
   default: return null;
  }
 }
```

198 EuropeanSwallow airSpeedVelocity

## class EuropeanSwallowDelegate...

```
get airSpeedVelocity() {return 35;}
```

## class EuropeanSwallow...

```
get airSpeedVelocity() {return this._speciesDelegate.airSpeedVelocity;}
```

airSpeedVelocity

## class Bird...

```
get airSpeedVelocity() {
  return this._speciesDelegate ? this._speciesDelegate.airSpeedVelocity : null;
}
```

```
class EuropeanSwallow extends Bird {
  get airSpeedVelocity() {
    return this._speciesDelegate.airSpeedVelocity;
  }
}
```

•••

```
function createBird(data) {
  switch (data.type) {
    case "EuropeanSwallow":
      return new EuropeanSwallow(data);
    case "AfricanSwallow":
      return new AfricanSwallow(data);
    case "NorweigianBlueParrot":
      return new NorwegianBlueParrot(data);
    default:
      return new Bird(data);
  }
}
```

AfricanSwallow data

## class AfricanSwallowDelegate...

```
constructor(data) {
  this._numberOfCoconuts = data.numberOfCoconuts;
}
```

class Bird…

```
selectSpeciesDelegate(data) {
 switch(data.type) {
  case 'EuropeanSwallow':
   return new EuropeanSwallowDelegate();
  case 'AfricanSwallow':
   return new AfricanSwallowDelegate(data);
  default: return null;
 }
}
```

198 airSpeedVelocity

## class AfricanSwallowDelegate...

```
get airSpeedVelocity() {
  return 40 - 2 * this._numberOfCoconuts;
}
```

## class AfricanSwallow...

```
get airSpeedVelocity() {
  return this._speciesDelegate.airSpeedVelocity;
}
```

AfricanSwallow

```
class AfricanSwallow extends Bird {
  // all of the body ...
}

function createBird(data) {
  switch (data.type) {
    case "AfricanSwallow":
      return new AfricanSwallow(data);
    case "NorweigianBlueParrot":
      return new NorwegianBlueParrot(data);
    default:
      return new Bird(data);
  }
}
```

NorwegianBlueParrot airSpeed Velocity

## class Bird...

```
selectSpeciesDelegate(data) {
 switch(data.type) {
  case 'EuropeanSwallow':
   return new EuropeanSwallowDelegate();
  case 'AfricanSwallow':
   return new AfricanSwallowDelegate(data);
  case 'NorweigianBlueParrot':
   return new NorwegianBlueParrotDelegate(data);
  default: return null;
 }
}
```

## class NorwegianBlueParrotDelegate...

```
constructor(data) {
  this._voltage = data.voltage;
  this._isNailed = data.isNailed;
}
get airSpeedVelocity() {
  return (this._isNailed) ? 0 : 10 + this._voltage / 10;
}
```

NorwegianBlueParrot plumage 198 plumage Bird

## class NorwegianBlueParrot...

```
get plumage() {
  return this._speciesDelegate.plumage;
}
```

## class NorwegianBlueParrotDelegate...

```
get plumage() {
 if (this._voltage > 100) return "scorched";
 else return this._bird._plumage || "beautiful";
}

constructor(data, bird) {
 this._bird = bird;
 this._voltage = data.voltage;
 this._isNailed = data.isNailed;
}
```

## class Bird...

```
selectSpeciesDelegate(data) {
 switch(data.type) {
  case 'EuropeanSwallow':
   return new EuropeanSwallowDelegate();
  case 'AfricanSwallow':
   return new AfricanSwallowDelegate(data);
  case 'NorweigianBlueParrot':
   return new NorwegianBlueParrotDelegate(data, this);
  default: return null;
 }
}
```

plumage plumage

## class Bird...

```
get plumage() {
  if (this._speciesDelegate)
    return this._speciesDelegate.plumage;
  else
    return this._plumage || "average";
}
```

## class Bird...

```
get plumage() {
  if (this._speciesDelegate instanceof NorwegianBlueParrotDelegate)
    return this._speciesDelegate.plumage;
  else
    return this._plumage || "average";
}
```

## class Bird...

```
get plumage() {
  if (this._speciesDelegate)
    return this._speciesDelegate.plumage;
  else
    return this._plumage || "average";
}
```

## class EuropeanSwallowDelegate...

```
get plumage() {
  return this._bird._plumage || "average";
}
```

## class AfricanSwallowDelegate...

```
get plumage() {
  return this._bird._plumage || "average";
}
```

plumage “ ” _bird

—— 375

```
class SpeciesDelegate {
 constructor(data, bird) {
  this._bird = bird;
 }
 get plumage() {
  return this._bird._plumage || "average";
 }

class EuropeanSwallowDelegate extends SpeciesDelegate {

class AfricanSwallowDelegate extends SpeciesDelegate {
 constructor(data, bird) {
  super(data,bird);
  this._numberOfCoconuts = data.numberOfCoconuts;
}

class NorwegianBlueParrotDelegate extends SpeciesDelegate {
 constructor(data, bird) {
  super(data, bird);
  this._voltage = data.voltage;
  this._isNailed = data.isNailed;
}
```

SpeciesDelegate Bird SpeciesDelegate

## class Bird...

```
selectSpeciesDelegate(data) {
 switch(data.type) {
  case 'EuropeanSwallow':
   return new EuropeanSwallowDelegate(data, this);
  case 'AfricanSwallow':
   return new AfricanSwallowDelegate(data, this);
  case 'NorweigianBlueParrot':
   return new NorwegianBlueParrotDelegate(data, this);
  default: return new SpeciesDelegate(data, this);
 }
}
// rest of bird's code...

get plumage() {return this._speciesDelegate.plumage;}

get airSpeedVelocity() {return this._speciesDelegate.airSpeedVelocity;}
```

## class SpeciesDelegate...

```
get airSpeedVelocity() {return null;}
```

Bird SpeciesDelegate Bird

```
function createBird(data) {
 return new Bird(data);
}
class Bird {
 constructor(data) {
  this._name = data.name;
  this._plumage = data.plumage;
  this._speciesDelegate = this.selectSpeciesDelegate(data);
 }
 get name()    {return this._name;}
 get plumage() {return this._speciesDelegate.plumage;}
 get airSpeedVelocity() {return this._speciesDelegate.airSpeedVelocity;}

 selectSpeciesDelegate(data) {
  switch(data.type) {
   case 'EuropeanSwallow':
    return new EuropeanSwallowDelegate(data, this);
   case 'AfricanSwallow':
    return new AfricanSwallowDelegate(data, this);
   case 'NorweigianBlueParrot':
    return new NorwegianBlueParrotDelegate(data, this);
   default: return new SpeciesDelegate(data, this);
  }
 }
 // rest of bird's code...
}

class SpeciesDelegate {
 constructor(data, bird) {
  this._bird = bird;
 }
 get plumage() {
  return this._bird._plumage || "average";
 }
 get airSpeedVelocity() {return null;}
}

class EuropeanSwallowDelegate extends SpeciesDelegate {
 get airSpeedVelocity() {return 35;}
}

class AfricanSwallowDelegate extends SpeciesDelegate {
 constructor(data, bird) {
  super(data,bird);
  this._numberOfCoconuts = data.numberOfCoconuts;
 }
 get airSpeedVelocity() {
  return 40 - 2 * this._numberOfCoconuts;
 }
}
```

```
class NorwegianBlueParrotDelegate extends SpeciesDelegate {
 constructor(data, bird) {
  super(data, bird);
  this._voltage = data.voltage;
  this._isNailed = data.isNailed;
 }
 get airSpeedVelocity() {
  return (this._isNailed) ? 0 : 10 + this._voltage / 10;
 }
 get plumage() {
  if (this._voltage > 100) return "scorched";
  else return this._bird._plumage || "beautiful";
 }
}
```

SpeciesDelegate Bird

Bird

“ ” Booking

“ ” “ ”____

# 12.11 Replace Superclass with Delegate

Replace Inheritance with Delegation

```
class List {...}
class Stack extends List {...}


class Stack {
  constructor() {
    this._storage = new List();
  }
}
class List {...}
```

stack list

____

“ ” car model

car VIN

“ ” type-instance homonym [mf-tih]

—— “

”

/

scroll catalog ID title

tag

## class CatalogItem...

```
constructor(id, title, tags) {
 this._id = id;
 this._title = title;
 this._tags = tags;
}

get id() {return this._id;}
get title() {return this._title;}
hasTag(arg) {return this._tags.includes(arg);}
```

Scroll CatalogItem “ ”

## class Scroll extends CatalogItem...

```
constructor(id, title, tags, dateLastCleaned) {
 super(id, title, tags);
 this._lastCleaned = dateLastCleaned;
}

needsCleaning(targetDate) {
 const threshold = this.hasTag("revered") ? 700 : 1500;
 return this.daysSinceLastCleaning(targetDate) > threshold ;
}
daysSinceLastCleaning(targetDate) {
 return this._lastCleaned.until(targetDate, ChronoUnit.DAYS);
}
```

“ ”

“ ” “ ”

“ ” “ ”

Scroll CatalogItem

## class Scroll extends CatalogItem...

```
constructor(id, title, tags, dateLastCleaned) {
  super(id, title, tags);
  this._catalogItem = new CatalogItem(id, title, tags);
  this._lastCleaned = dateLastCleaned;
}
```

## class Scroll...

```
get id() {return this._catalogItem.id;}
get title() {return this._catalogItem.title;}
hasTag(aString) {return this._catalogItem.hasTag(aString);}
```

Scroll CatalogItem

```
class Scroll extends CatalogItem{
  constructor(id, title, tags, dateLastCleaned) {
    super(id, title, tags);
    this._catalogItem = new CatalogItem(id, title, tags);
    this._lastCleaned = dateLastCleaned;
  }
```

CatalogItem Scroll Scroll CatalogItem
6 6 256

256 Scroll CatalogItem id ID
CatalogItem ID ID ID Scroll id Scroll
CatalogItem id

## class Scroll...

```
constructor(id, title, tags, dateLastCleaned) {
  this._id = id;
  this._catalogItem = new CatalogItem(null, title, tags);
  this._lastCleaned = dateLastCleaned;
}

get id() {return this._id;}
```

null ID ID

Scroll

•••

```
const scrolls = aDocument
  .map(record => new Scroll(record.id,
              record.catalogData.title,
              record.catalogData.tags,
              LocalDate.parse(record.lastCleaned)));
```

256 repository CatalogItem
ID ID Scroll
124 ID Scroll

•••

```
const scrolls = aDocument
  .map(record => new Scroll(record.id,
              record.catalogData.title,
              record.catalogData.tags,
              LocalDate.parse(record.lastCleaned),
              record.catalogData.id,
              catalog));
```

## class Scroll...

```
  constructor(id, title, tags, dateLastCleaned, catalogID, catalog) {
 this._id = id;
 this._catalogItem = new CatalogItem(null, title, tags);
 this._lastCleaned = dateLastCleaned;
}
```

Scroll catalogID CatalogItem CatalogItem

## class Scroll...

```
constructor(id, title, tags, dateLastCleaned, catalogID, catalog) {
  this._id = id;
  this._catalogItem = catalog.get(catalogID);
  this._lastCleaned = dateLastCleaned;
}
```

Scroll title tags 124

...

```
const scrolls = aDocument
  .map(record => new Scroll(record.id,
              record.catalogData.title,
              record.catalogData.tags,
              LocalDate.parse(record.lastCleaned),
              record.catalogData.id,
              catalog));
```

## class Scroll...

```
  constructor(id, title, tags, dateLastCleaned, catalogID, catalog) {
 this._id = id;
 this._catalogItem = catalog.get(catalogID);
 this._lastCleaned = dateLastCleaned;
}
```