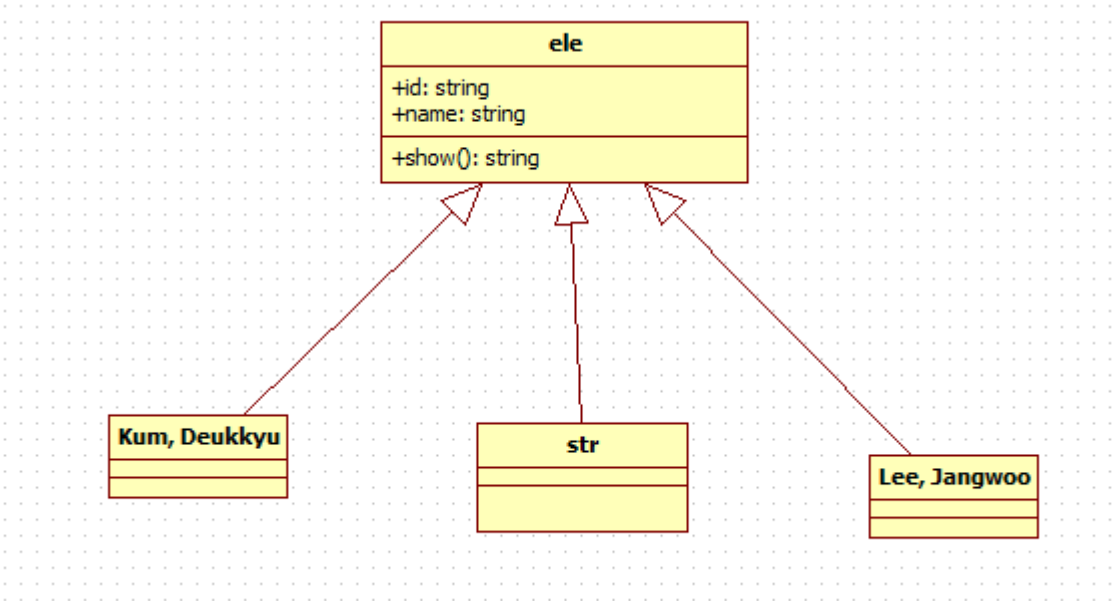
网页组件结构的图形化表示和zen-coding简化码表示

网页组件结构的图形化表示

> 这种图示方法是笔者（巽星子）自创的，早些年在自学VB6.0的自定义控件、类和组件化编程时使用过。甚至没有一个固定的名称可以叫它。
>
> 这种图示方法可以图形化的表示一些HTML标签代码结构或js对象、PHP类结构，乃至新生的一些前端框架的组件对象（如Vue.js以及它的第三方UI组件库Element）
>
> 你说这种图示有什么作用，我觉得它最好的用途是作为对象结构的描述，可以简单的描述一个对象的属性、事件、方法以及对象之间嵌套关系。

和UML类图的区别

UML类图可以描述类的成员（属性和方法等）和类之间的继承关系（最常见的关系），却难以有效的描述像HTML标签这样的嵌套结构或复杂js对象的结构（对象的成员是新的对象之类）。

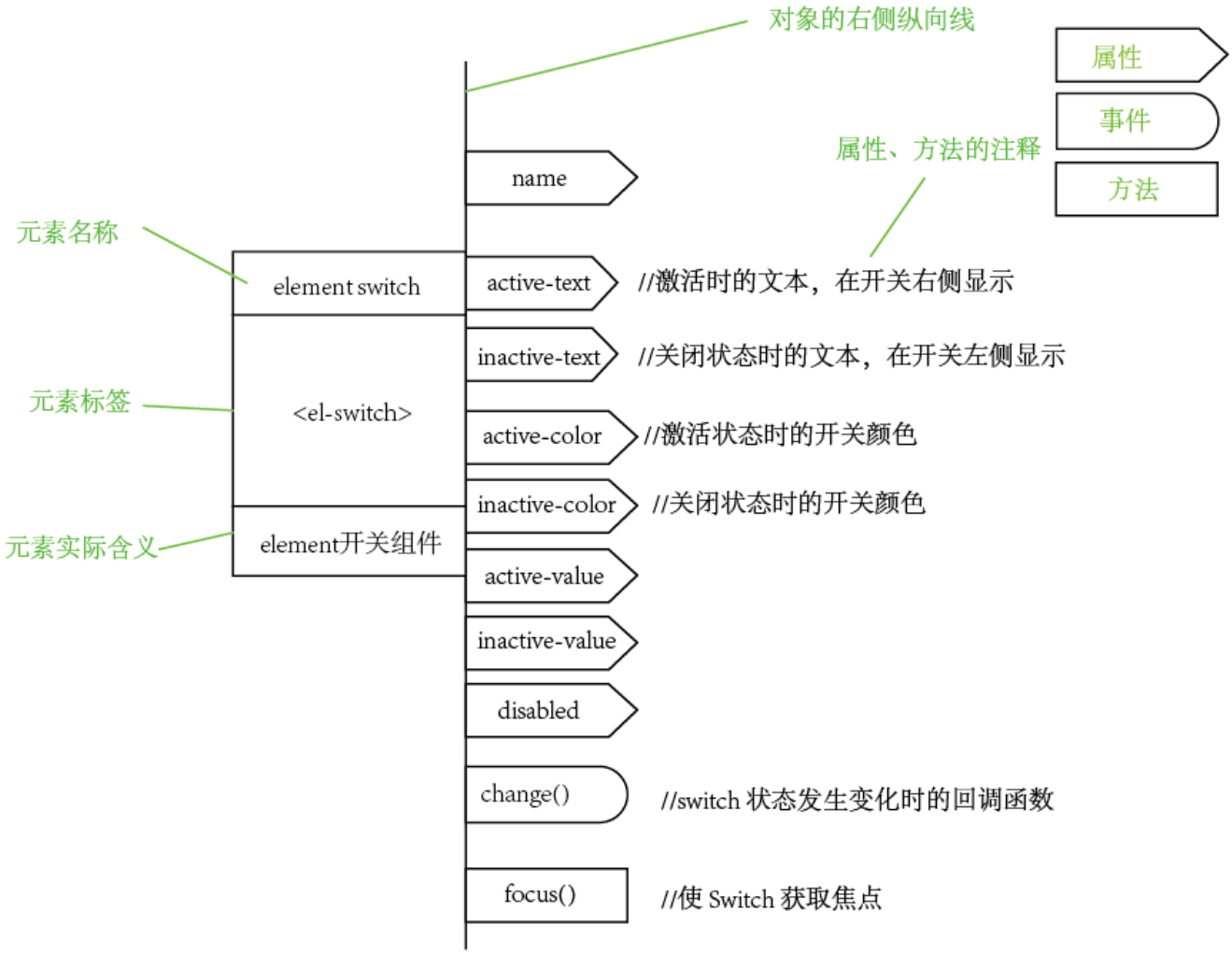


优点

1. 易于绘制：
    1. 本篇插图使用Adobe illustrator(AI)绘制，绘制过程基本没什么困难；
    2. 在方格本或A4纸等草稿上手绘也特别简单。
2. 有限的图例：这一点也保证了绘制的简单。
3. 理论上无限的向下嵌套层次：HTML结构代码块或JS复杂对象等理论上可以无限嵌套，而本图示方法也可以理论上无限的绘制下去。
4. 应用场景：

1. 技术博客或文档：可搭配Markdown或wiki；
2. 编程类图书：包括速查手册在内，其实可以将这种图示广泛的使用起来，用于程序员或爱好者查询，比起大段的文字描述和四四方方的表格，或许更容易使用。
3. 设计自己的框架或代码时使用：设计阶段就开始使用（促进设计思考形成），设计结束形成文档（用于速查）。

图例



对象：

- 对象由三行的矩形组成，中间一行宽大，上下两行窄
  - 第一行写元素的名称，通常写成"语言或库 元素或组件名"形式；
  - 第二行写HTML标签或js对象或其他语言类、对象的名称；
  - 第三行相当于是第一行的进一步解释，通常有中文语句；
- 对象的简化表示法：
  - 可以仅保留一个大矩形来书写元素的标签或对象名称等；

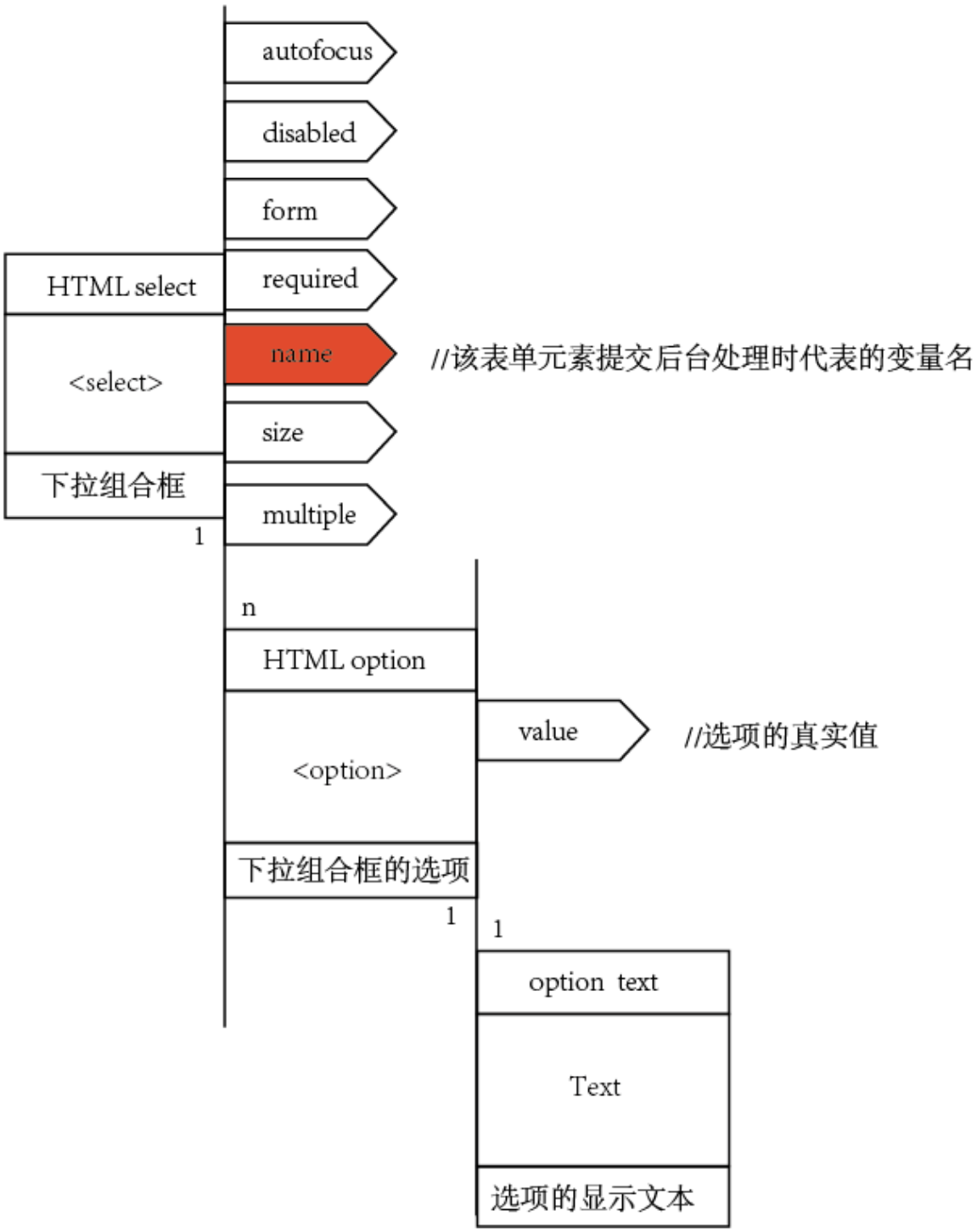
对象成员图例：

- 对象的属性用后部带尖角的块表示
- 父对象在左，子对象在右，并且子对象和父对象的属性、事件、方法等成员画在父对象的右侧纵向延伸线上

- 可以使用“1-1”、“1-n”等表示父对象与子对象之间的一对一和一对多关系

原生HTML元素的图形化表示

下面的结构图代表了HTML的原生表单元素select的结构代码组成。阐述了select、option标签之间的一对多包含关系，并且标出了两个标签各自的属性。



虽然像普通的HTML标签这样的大部分的代码编辑器和IDE提供了很好的语法提示，但是在一些特化的框架或代码面前，很多编辑器不一定能够提供很好的代码提示。

a.html

```html
<select name="dq" id="dq" multiple size="10">
    <option value="dq001" selected>青海</option>
    <option value="dq002">新疆</option>
    <option value="dq003">西藏</option>
</select>
```

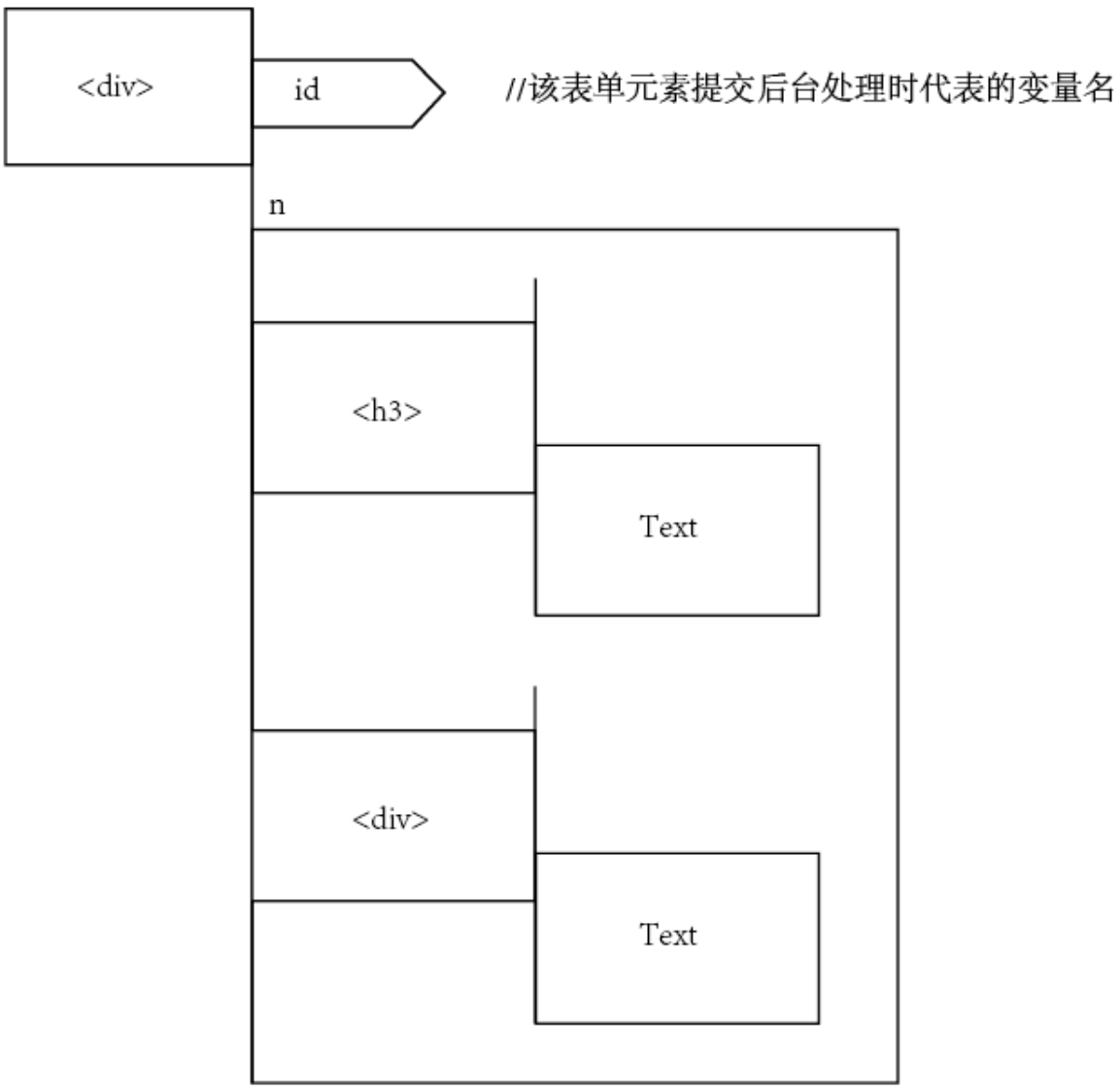
上图结构代码的zen-coding大致等价表示法：

```
select#dq[mutiple size=10]>option*3
```

- zen-coding表示法在HTML结构代码块的表示方面有一定的作用；
- 诸多代码编辑器支持zen-coding语法来快速书写生成HTML结构代码块；

Jquery UI组件结构代码图形化表示

下面代表Jquery UI手风琴面板组件的图形化表示。



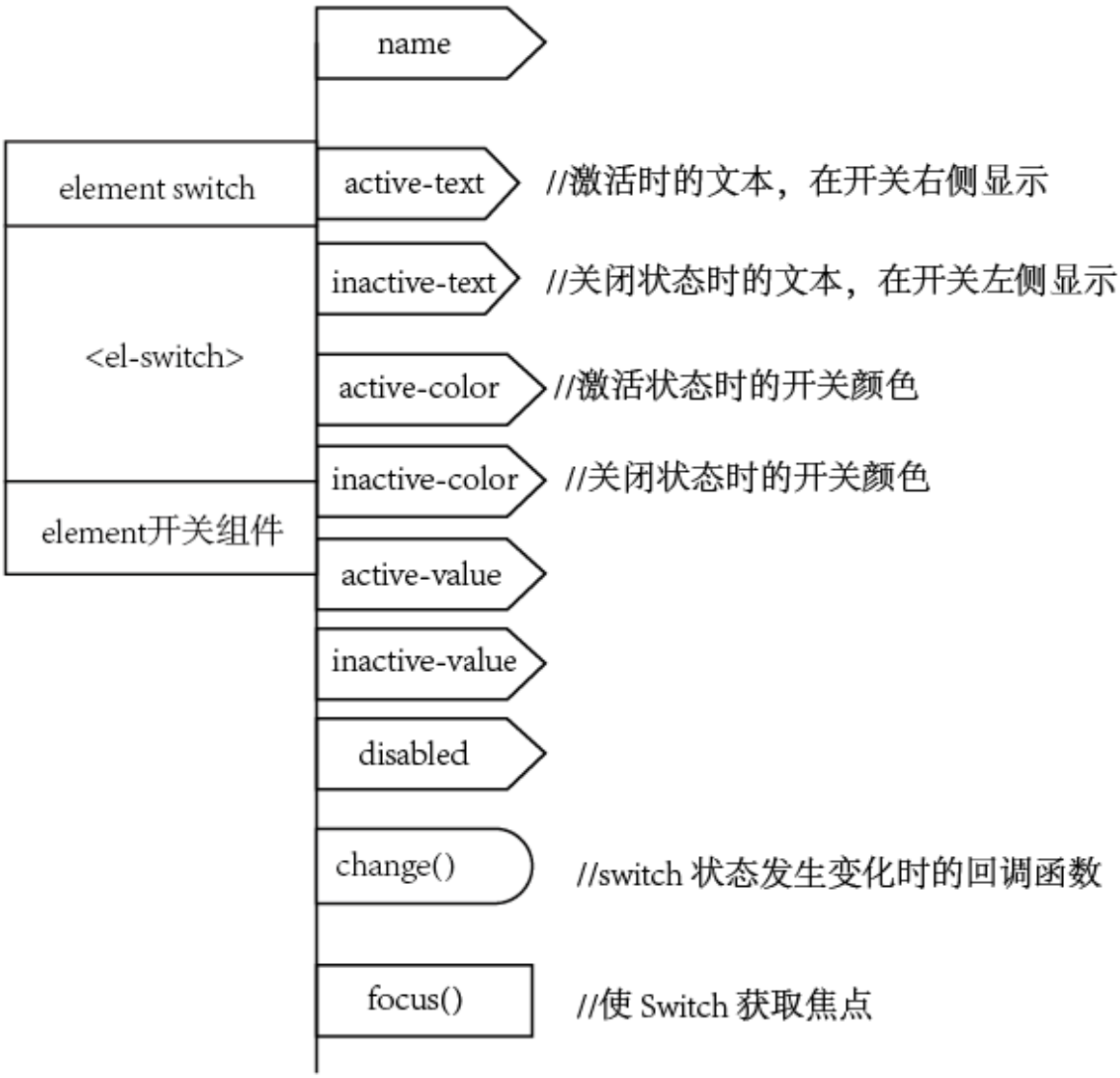
```
<div id="acc">
    <h3></h3>
    <div></div>
    <h3></h3>
    <div></div>
    <h3></h3>
    <div></div>
</div>
```

上图结构代码的zen-coding表示法：

```
div#acc>(h3+div)*3
```

Vue ElementUI组件的元素对象图形化表示

以el-switch组件为例，进行组件元素对象的图形化表示。



上面的图中引入了新的图例，比如说U形块代表了对象的事件，长方条代表对象方法。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
    <script src="https://cdn.bootcss.com/vue/2.6.10/vue.min.js"></script>
    <!-- 引入样式 -->
    <link rel="stylesheet" href="https://unpkg.com/element-ui/lib/theme-chalk/index.css">
    <!-- 引入组件库 -->
    <script src="https://unpkg.com/element-ui/lib/index.js"></script>
</head>
<body>
    <div id="app">
        <el-switch v-model="sw" @change="change()"></el-switch>
    </div>
<script>
    var v=new Vue({
        el:"#app",
        data:{
            sw:true
        },
        methods:{
            change:function(){
                //change事件处理
            }
        }
    });
</script>
</body>
</html>
```

本篇暂未描述JS对象或其他语言如类的结构，后续再述。

此系列文章发布于GeekerNote公众号，更多作为作者自己总结二次查阅使用，若能帮助到某些朋友，深感荣幸。或许尝试将系列文章发布至CSDN等技术帐号。