30 天精通RxJS (03)：Functional Programming 通用函式

Dec 18th, 2016 . 4 mins read

> 了解Functional Programming 的通用函式，能让我们写出更简洁的程式码，也能帮助我们学习RxJS。

这是【30天精通RxJS】的02篇，如果还没看过01篇可以往这边走： 30天精通RxJS (01)：认识RxJS

读者可能会很好奇，我们的主题是RxJS 为什么要特别讲Functional Programming 的通用函式呢？实际上，RxJS 核心的Observable 操作观念跟FP 的阵列操作是极为相近的，只学会以下几个基本的方法跟观念后，会让我们之后上手Observable 简单很多！

今天的程式码比较多，大家可以直接看影片！

ForEach

> forEach 是JavaScript 在ES5 后，原生就有支援的方法。

原本我们可能要透过for loop 取出阵列中的每一个元素

```
var arr = ['Jerry', 'Anna'];

for(var i = 0; i < arr.length; i++) {
        console.log(arr[i]);
}
```

现在可以直接透过阵列的forEach 取出每一个元素。

```
var arr = ['Jerry', 'Anna'];

arr.forEach(item => console.log(item));
```

Map

试着把newCourseList 每个元素的{ id, title } 塞到新的阵列idAndTitlePairs

```
var newCourseList = [
        {
                "id": 511021,
                "title": "React for Beginners",
                "coverPng": "https://res.cloudinary.com/dohtkyi84/image
                "rating": 5
        },
        {
                "id": 511022,
                "title": "Vue2 for Beginners",
                "coverPng": "https://res.cloudinary.com/dohtkyi84/image
                "rating": 5
        },
        {
                "id": 511023,
                "title": "Angular2 for Beginners",
                "coverPng": "https://res.cloudinary.com/dohtkyi84/image
                "rating": 5
        },
        {
                "id": 511024,
                "title": "Webpack for Beginners",
                "coverPng": "https://res.cloudinary.com/dohtkyi84/image
                "rating": 4
        }
], idAndTitle = [];

newCourseList.forEach((course) => {
        idAndTitle.push({ id: course.id, title: course.title });
});
```

上面我们练习到newCourseList 转换成一个新的阵列idAndTitlePairs，这个转换的过程其实就是两件事

- 遍历newCourseList 所有的元素
- 把每个元素的预期值给到新的阵列

把这个过程抽象化成一个方法map，以下是简化的基本思路：

1. 我们会让每个阵列都有一个map 方法
2. 这个方法会让使用者自订传入一个callback function
3. 这个callback function 会回传使用者预期的元素

> 虽然ES5 之后原生的JavaScript 阵列有map 方法了，但希望读者自我实做一次，能帮助理解。

```
// 我們希望每一個陣列都有 map 這個方法，所以我們在 Array.prototype 擴充 map fu
Array.prototype.map = function(callback) {
  var result = []; // map 最後一定會返回一個新陣列，所以我們先宣告一個新陣列

  this.forEach(function(element, index) {
          // this 就是呼叫 map 的陣列
          result.push(callback(element, index));
          // 執行使用者定義的 callback， callback 會回傳使用者預期的元素，所以
  })

  return result;
}
```

> 這裡用到了 JavaScript 的 prototype chain 以及 this 等觀念，可以看此影片了解！

到這裡我們就實作完成 map 的方法了，讓我們來試試這個方法吧！

```
var idAndTitle = newCourseList
                 .map((course) => {
                     return { id: course.id, title: course.title };
                 });
```

Filter

由于最近很红的Redux 使我能很好的举例，让大家了解什么是用参数保存状态。了解Redux 的开发者应该会知Redux 的状态是由各个reducer 所组成的，而每个reducer 的状态就是保存在参数中！

讓我們過濾出 rating 值是 5 的元素

```
var ratingIsFive = [];

newCourseList.forEach((course) => {
        if(course.rating === 5) {
                ratingIsFive.push(course);
        }
});
```

同樣的我們試著來簡化這個過程，首先在這個轉換的過程中，我們做了兩件事：

1. 遍歷 newCourseList 中的所有元素
2. 判斷元素是否符合條件，符合則加到新的陣列中

```
Array.prototype.filter = function(callback) {
        var result = [];
        this.forEach((item, index) => {
                if(callback(item, index))
                        result.push(item);
        });
        return result;
}
```

試試這個方法

```
var ratingIsFive = newCourseList
                   .filter((course) => course.rating === 5);
```

會發現我們的程式碼又變簡單了，接著我們試著把 filter, map 串起來。

```
    .map(course => course.title);
```

ConcatAll

有時候我們會遇到組出一個二維陣列，但我們希望陣列是一維的，問題如下：

假如我們要取出 courseLists 中所有 rating 為 5 的課程，這時可能就會用到兩個 forEach

```
var user = {
  id: 888,
  name: 'JerryHong',
  courseLists: [{
    "name": "My Courses",
    "courses": [{
      "id": 511019,
      "title": "React for Beginners",
      "coverPng": "https://res.cloudinary.com/dohtkyi84/image/upload/v1
      "tags": [{ id: 1, name: "JavaScript" }],
      "rating": 5
    }, {
      "id": 511020,
      "title": "Front-End automat workflow",
      "coverPng": "https://res.cloudinary.com/dohtkyi84/image/upload/v1
      "tags": [{ "id": 2, "name": "gulp" }, { "id": 3, "name": "webpack
      "rating": 4
    }]
  }, {
    "name": "New Release",
    "courses": [{
      "id": 511022,
      "title": "Vue2 for Beginners",
      "coverPng": "https://res.cloudinary.com/dohtkyi84/image/upload/v1
      "tags": [{ id: 1, name: "JavaScript" }],
      "rating": 5
    }, {
      "id": 511023,
```

```
    }]
  }]
};

var allCourseIds = [];

user.courseLists.forEach(list => {
  list.courses
    .filter(item => item.rating === 5)
    .forEach(item => {
      allCourseIds.push(item)
    })
})
```

可以看到上面的程式碼，我們用了較為低階的操作來解決這個問題，我們剛剛已經試著用抽象化的方式實作了 map 跟 filter，那我們同樣也能夠定義一個方法用來 攤平二維陣列。

讓我們來加入一個 concatAll 方法來簡化這段程式碼吧！ concatAll 要做的事情很簡單，就是把一個二維陣列轉成一維。

```
Array.prototype.concatAll = function() {
  var result = [];

  // 用 apply 完成
  this.forEach((array) => {
    result.push.apply(result, array);
  });

  // 用兩個 forEach 完成
  // this.forEach((array) => {
  //   array.forEach(item => {
  //     result.push(item)
  //   })
  // });

  // 用 ES6 spread 完成
  // this.forEach((array) => {
```

```
};
```

同樣的我們用前面定要好的 courseLists 來試試 concatAll 吧！

```
var allCourseIds = user.courseLists.map(list => {
        return list.courses.filter(course => course.rating === 5)
}).concatAll()
```

這邊出一個比較難的題目，大家可以想想看要怎麼解

```
var courseLists = [{
  "name": "My Courses",
  "courses": [{
    "id": 511019,
    "title": "React for Beginners",
    "covers": [{
      width: 150,
      height: 200,
      url: "http://placeimg.com/150/200/tech"
    }, {
      width: 200,
      height: 200,
      url: "http://placeimg.com/200/200/tech"
    }, {
      width: 300,
      height: 200,
      url: "http://placeimg.com/300/200/tech"
    }],
    "tags": [{
      id: 1,
      name: "JavaScript"
    }],
    "rating": 5
  }, {
    "id": 511020,
    "title": "Front-End automat workflow",
    "covers": [{
      width: 150
```





```
        width: 200,
        height: 200,
        url: "http://placeimg.com/200/200/arch"
      }, {
        width: 300,
        height: 200,
        url: "http://placeimg.com/300/200/arch"
      }],
      "tags": [{
        "id": 2,
        "name": "gulp"
      }, {
        "id": 3,
        "name": "webpack"
      }],
      "rating": 5
    }]
  }, {
    "name": "New Release",
    "courses": [{
      "id": 511022,
      "title": "Vue2 for Beginners",
      "covers": [{
        width: 150,
        height: 200,
        url: "http://placeimg.com/150/200/nature"
      }, {
        width: 200,
        height: 200,
        url: "http://placeimg.com/200/200/nature"
      }, {
        width: 300,
        height: 200,
        url: "http://placeimg.com/300/200/nature"
      }],
      "tags": [{
        id: 1,
        name: "JavaScript"
```





```
    "id": 511023,
    "title": "Angular2 for Beginners",
    "covers": [{
      width: 150,
      height: 200,
      url: "http://placeimg.com/150/200/people"
    }, {
      width: 200,
      height: 200,
      url: "http://placeimg.com/200/200/people"
    }, {
      width: 300,
      height: 200,
      url: "http://placeimg.com/300/200/people"
    }],
    "tags": [{
      id: 1,
      name: "JavaScript"
    }],
    "rating": 5
  }]
}];

/*
var result = courseList
不得直接使用索引 covers[0]，請用 concatAll, map, filter, forEach 完成
result 結果為 [
    {
      id: 511019,
      title: "React for Beginners",
      cover: "http://placeimg.com/150/200/tech"
    }, {
      id: 511020,
      title: "Front-End automat workflow",
      cover: "http://placeimg.com/150/200/arch"
    }, {
      id: 511022,
      title: "Vue2 for Beginners"
```





```
      title: "Angular2 for Beginners",
      cover: "http://placeimg.com/150/200/people"
    },
  ]
 */
```

練習連結： JSBin | JSFiddle

這題有點難，大家可以想想看，我把答案寫在這裡了！

如果大家還想做更多的練習可以到這個連結： http://reactivex.io/learnrx/

> 這個連結是 Jafar 大神為他的 RxJS workshop 所做的練習網站！

今日小結

今天讲了FP 操作阵列的三个通用函式forEach, map, filter，以及我们自己定义的一个方法叫concatAll。这几天我们把学习RxJS 的前置观念跟知识基本上都讲完了，明天我们就开始进入RxJS 的重点核心Observable 啰！

Tags

JavaScript RxJS Functional Programming RxJS 30 Days

Prev Next