Java8 lambda 和 Optional 的理解

作者：tiangao
原文链接：https://ld246.com/article/1524809579239
来源网站：链滴

用了这么久的 Java8, 现在才觉得对于 lambda 和 Optional 有了一点理解。以前一直不理解 Optional 的意义，不就是判断控制吗? if（xxx != null） 也没什么妨碍理解的嘛，多清晰。

然后今天有段代码是这样：

```
if (queryRequest.getBrandId() != null && queryRequest.getBrandId() > 0) {
    paramMap.put("brandId", queryRequest.getBrandId());
}
if (queryRequest.getCusineId() != null && queryRequest.getCusineId() > 0) {
    paramMap.put("cuisineId", queryRequest.getCusineId());
}
if (queryRequest.getLocalId() != null && queryRequest.getLocalId() > 0) {
    paramMap.put("localId", queryRequest.getLocalId());
}
if (queryRequest.getFaceId() != null && queryRequest.getFaceId() > 0) {
    paramMap.put("faceId", queryRequest.getFaceId());
}
if (queryRequest.getNow() != null && queryRequest.getNow() > 0) {
    paramMap.put("now", queryRequest.getNow());
}
if (queryRequest.getPice() != null && queryRequest.getPice() > 0) {
    paramMap.put("pice", queryRequest.getPice());
}
```

用 Optional 和 lambda 重构后变成了这样...

```
queryRequest.getBrandId().filter(id -> id > 0).ifPresent(id -> paramMap.put("brandId", id +""))


queryRequest.getCuisineId().filter(id -> id > 0).ifPresent(id -> paramMap.put("cuisineId", id +
"));

queryRequest.getLocalId().filter(id -> id > 0).ifPresent(id -> paramMap.put("localId", id +""));

queryRequest.getFaceId().filter(id -> id > 0).ifPresent(id -> paramMap.put("faceId", id +""));

queryRequest.getNow().filter(s -> !s.isEmpty()).ifPresent(v -> paramMap.put("now", v));

queryRequest.getPice().filter(p -> p > 0).ifPresent(p -> paramMap.put("foodIds", p +""));
```

当这样的判断拼参数有多达二三十处的时候，重构完真是看起来无比清爽。

还有一个计算然后返回参数的例子，如下，你们会喜欢哪一种写法呢?

```
package com.ctrip.gs.demo.web;

import java.util.Arrays;
import java.util.List;
import java.util.Optional;

public class Main {

    public static void main(String[] args) {
        QueryCondition queryCondition = new QueryCondition();
```

```
        System.out.println("0 -- " + Integer.toBinaryString(calByLambda(queryCondition)));

        List types = Arrays.asList(BinType.One, BinType.Three);
        queryCondition.setOptTypes(types);
        queryCondition.setTypes(types);
        System.out.println("2 lambda -- " + Integer.toBinaryString(calByLambda(queryCondition)
);
        System.out.println("2 normal -- " + Integer.toBinaryString(calByNormal(queryCondition)))


        types = Arrays.asList(BinType.Two, BinType.Three, BinType.Four);
        queryCondition.setOptTypes(types);
        queryCondition.setTypes(types);
        System.out.println("3 lambda -- " + Integer.toBinaryString(calByLambda(queryCondition)
);
        System.out.println("3 normal -- " + Integer.toBinaryString(calByNormal(queryCondition)))

    }

    private static Integer calByLambda(QueryCondition queryCondition) {
        return queryCondition.getOptTypes()
                .map(list -> list.stream()
                .map(BinType::getValue).reduce((a, b) -> a | b).orElse(0))
                .orElse(0);
    }

    private static Integer calByNormal(QueryCondition queryCondition) {
            int result = 0;
        List types = queryCondition.getTypes();
        if (types != null) {
            for (BinType type : types) {
                result = result | type.getValue();
            }
        }
        return result;
    }
}

class QueryCondition {
    private Optional<List<BinType>> optTypes = Optional.empty();

    private List types;

    public void setTypes(List types) {
        this.types = types;
    }

    public List getTypes() {
        return types;
    }

    public Optional<List<BinType>> getOptTypes() {
        return optTypes;
    }
```

```
    public void setOptTypes(List optTypes) {
        this.optTypes = Optional.of(optTypes);
    }
}

enum BinType {
    One(1), Two(1 << 1), Three(1 << 2), Four(1 << 3);

    int value;

    BinType(int value) {
        this.value = value;
    }

    public int getValue() {
        return value;
    }
}
```

原文链接：Java8 lambda 和 Optional 的理解