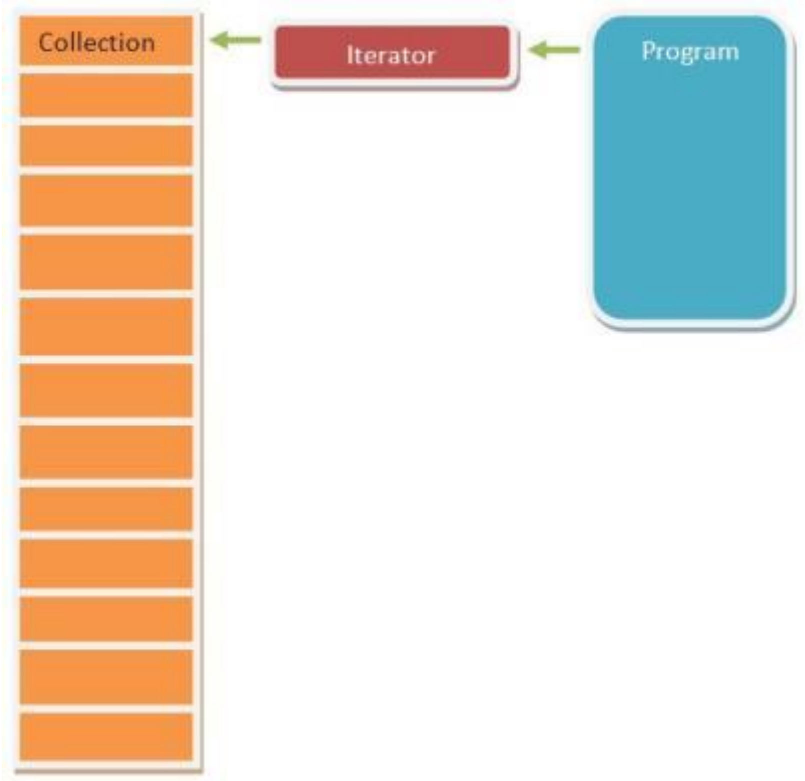
迭代器模式Iterator

波波老师~研发总监/资深架构师

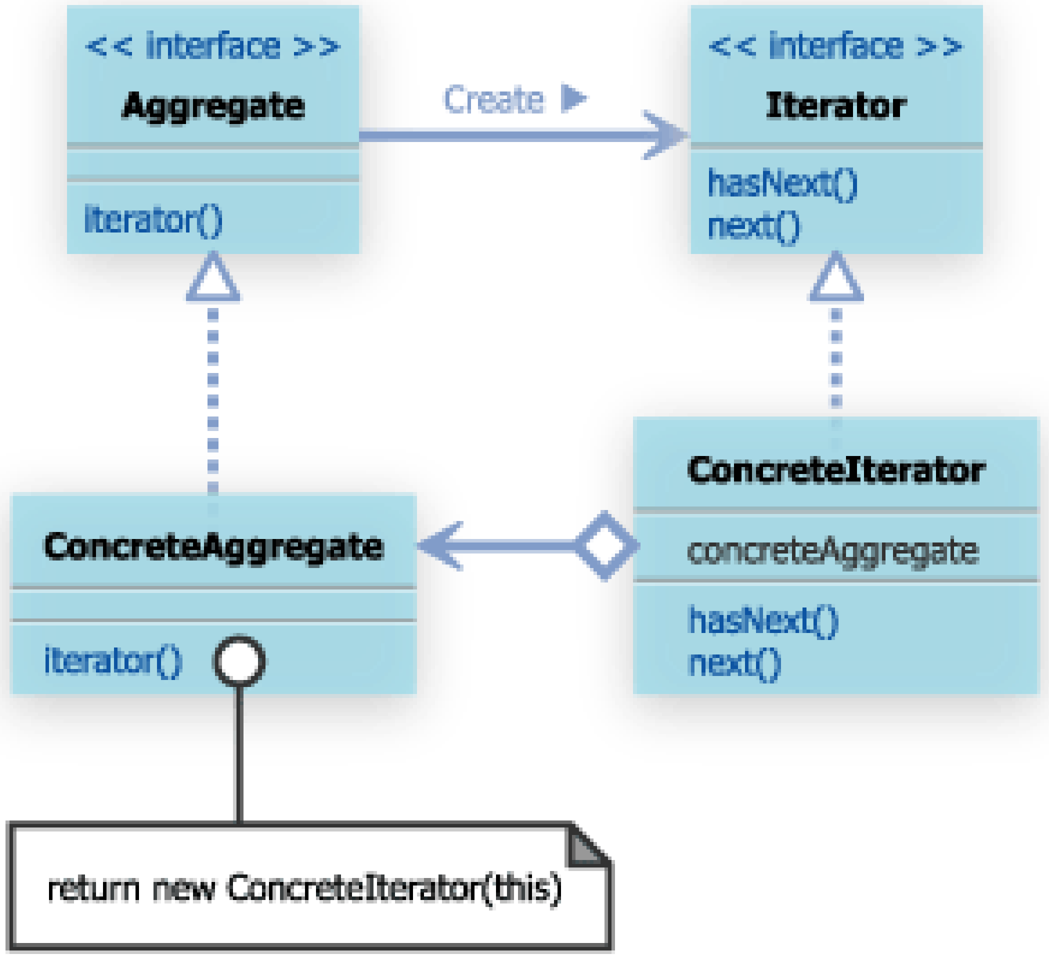
波波微课
spring2go.com

定义

- 迭代器提供一种遍历集合中元素的方式，不需要暴露元素的底层表示
- 集合和集合遍历逻辑解耦
- 客户端和集合遍历逻辑解耦

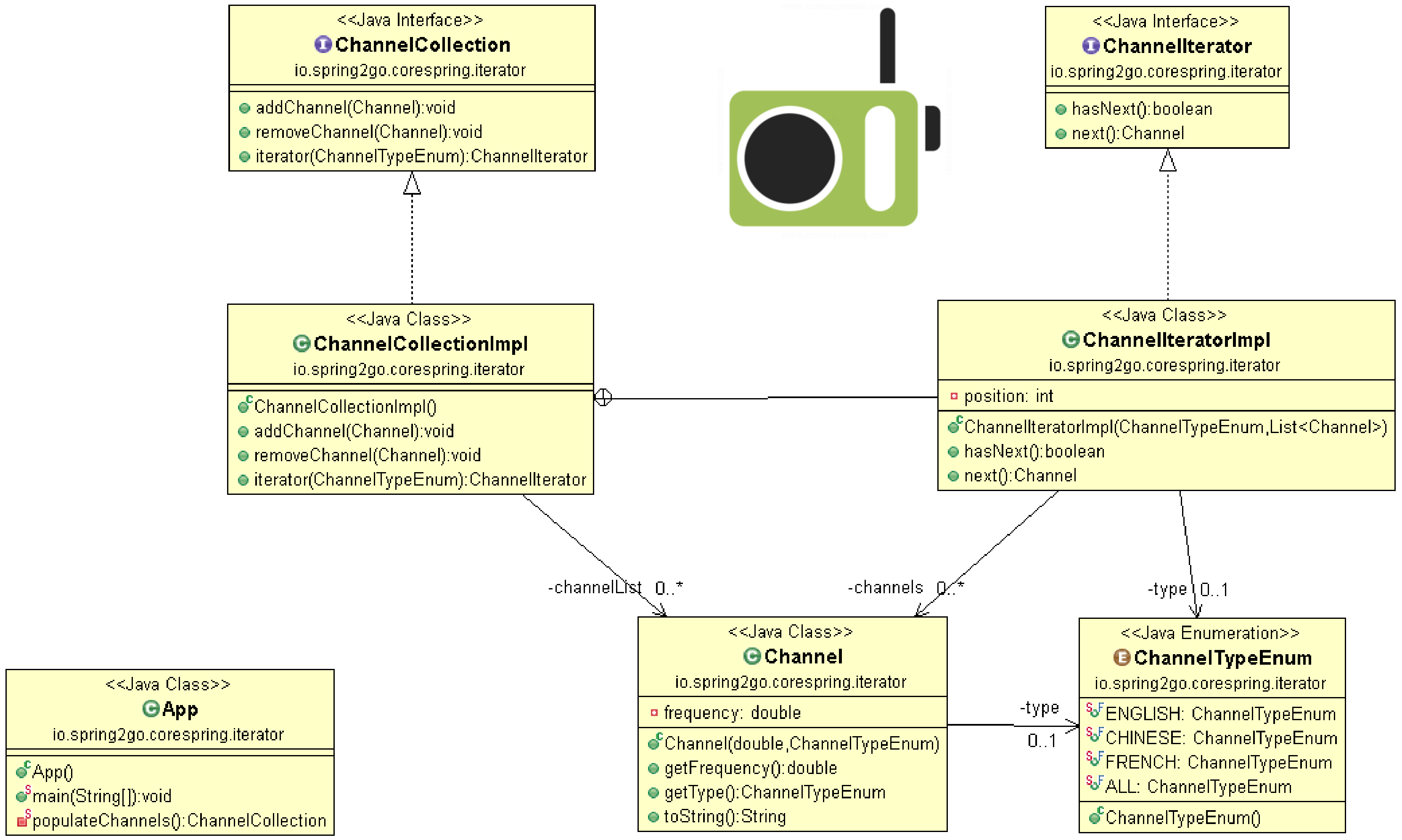


关系图

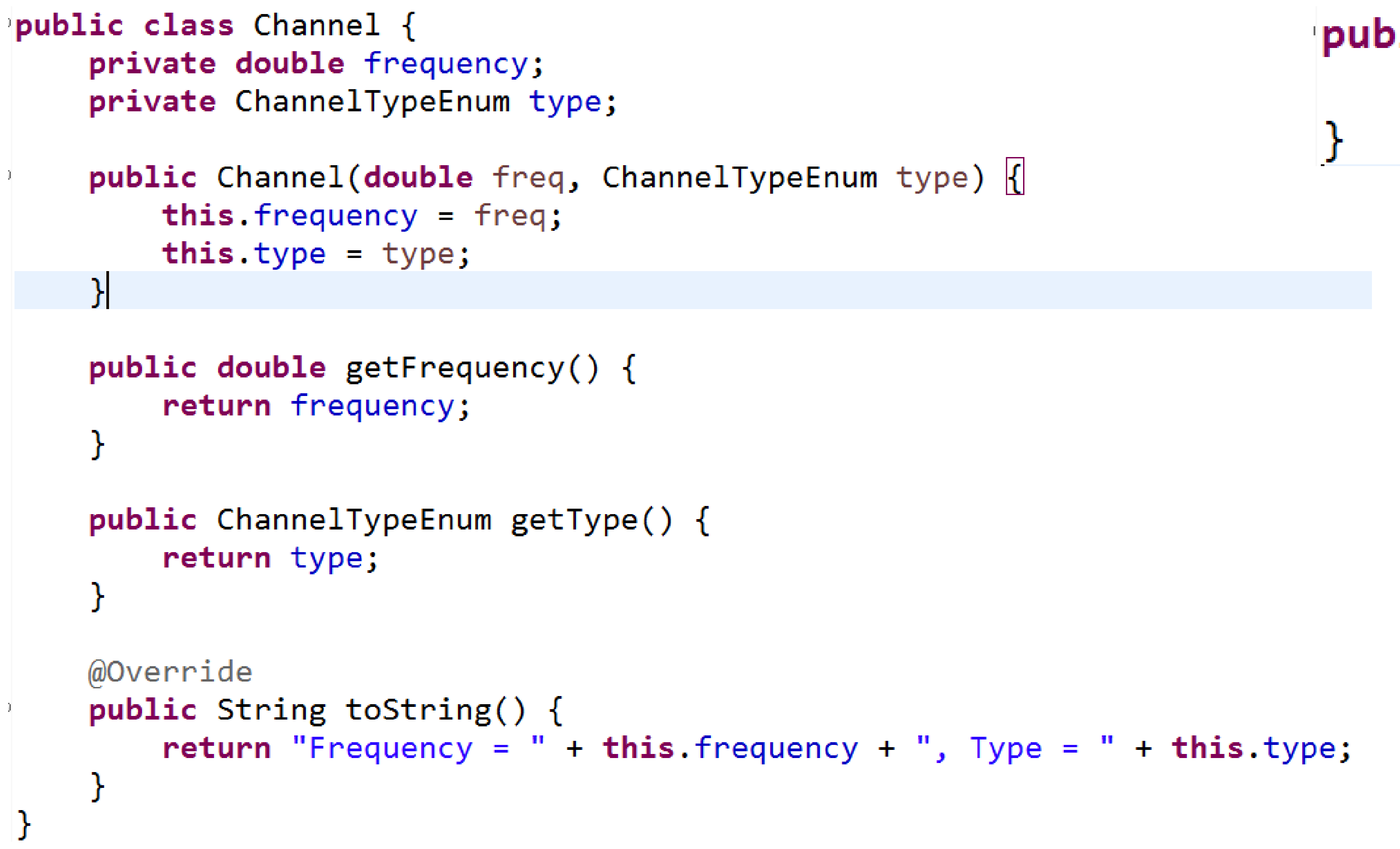


<< interface >>
Aggregate
iterator()
Create ▶
<< interface >>
Iterator
hasNext()
next()
ConcreteIterator
concreteAggregate
hasNext()
next()
ConcreteAggregate
iterator()
return new ConcreteIterator(this)

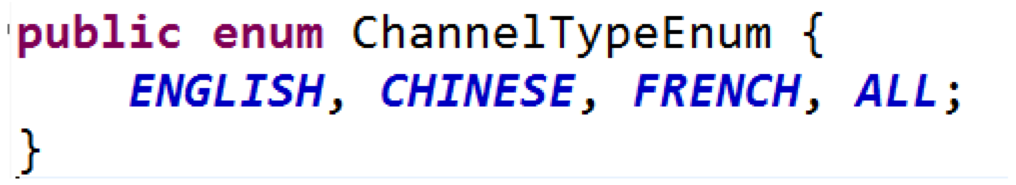
案例~Radio Channel Iterator



代码~Channel & ChannelTypeEnum

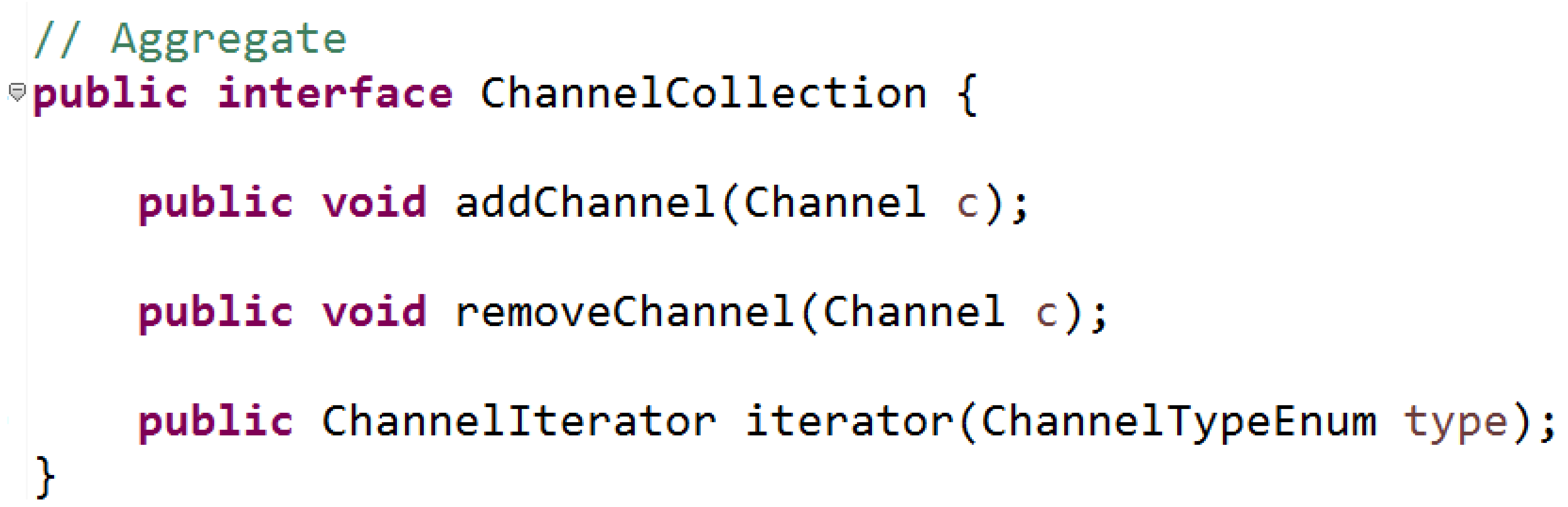


```java
public class Channel {
    private double frequency;
    private ChannelTypeEnum type;

    public Channel(double freq, ChannelTypeEnum type) {
        this.frequency = freq;
        this.type = type;
    }

    public double getFrequency() {
        return frequency;
    }

    public ChannelTypeEnum getType() {
        return type;
    }

    @Override
    public String toString() {
        return "Frequency = " + this.frequency + ", Type = " + this.type;
    }
}
```



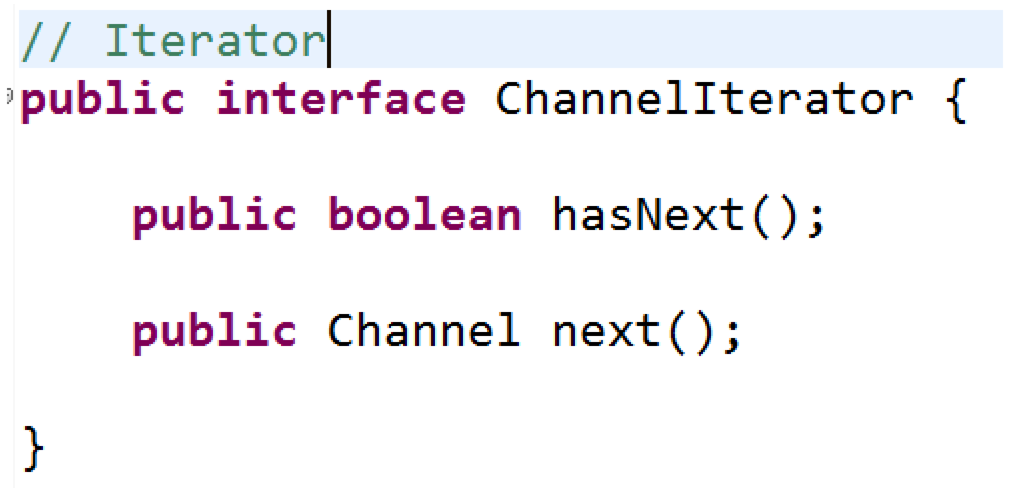
```java
public enum ChannelTypeEnum {
    ENGLISH, CHINESE, FRENCH, ALL;
}
```

代码~ ChannelCollection



```java
// Aggregate
public interface ChannelCollection {

    public void addChannel(Channel c);

    public void removeChannel(Channel c);

    public ChannelIterator iterator(ChannelTypeEnum type);
}
```

代码~ChannelIterator



```
// Iterator
public interface ChannelIterator {

    public boolean hasNext();

    public Channel next();

}
```

代码~ChannelCollectionImpl

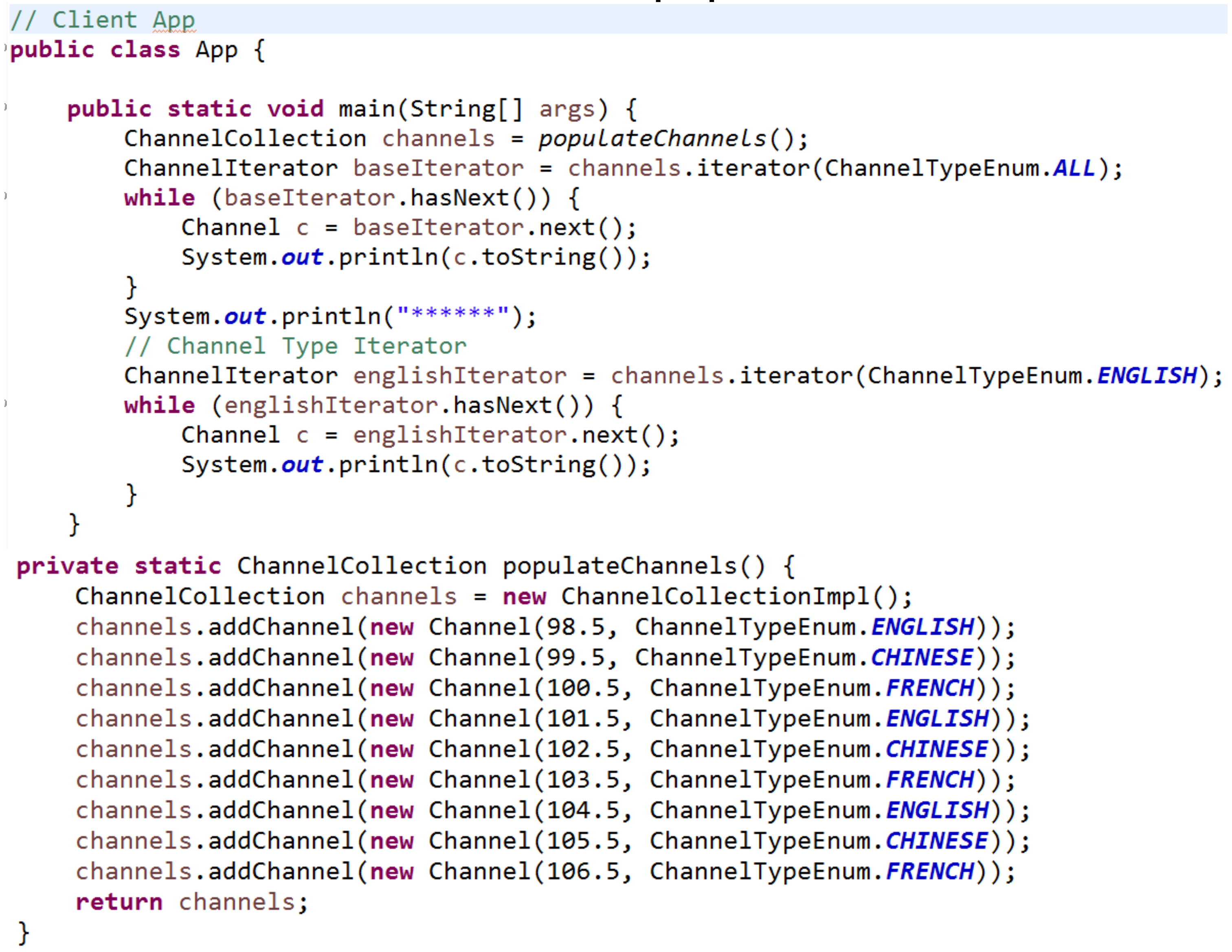
```java
// ConcreteAggregate
public class ChannelCollectionImpl implements ChannelCollection {

    private List<Channel> channelList;

    public ChannelCollectionImpl() {
        channelList = new ArrayList<>();
    }

    @Override
    public void addChannel(Channel c) {
        this.channelList.add(c);
    }

    @Override
    public void removeChannel(Channel c) {
        this.channelList.remove(c);
    }

    @Override
    public ChannelIterator iterator(ChannelTypeEnum type) {
        return new ChannelIteratorImpl(type, this.channelList);
    }
```

代码~ChannelIteratorImpl

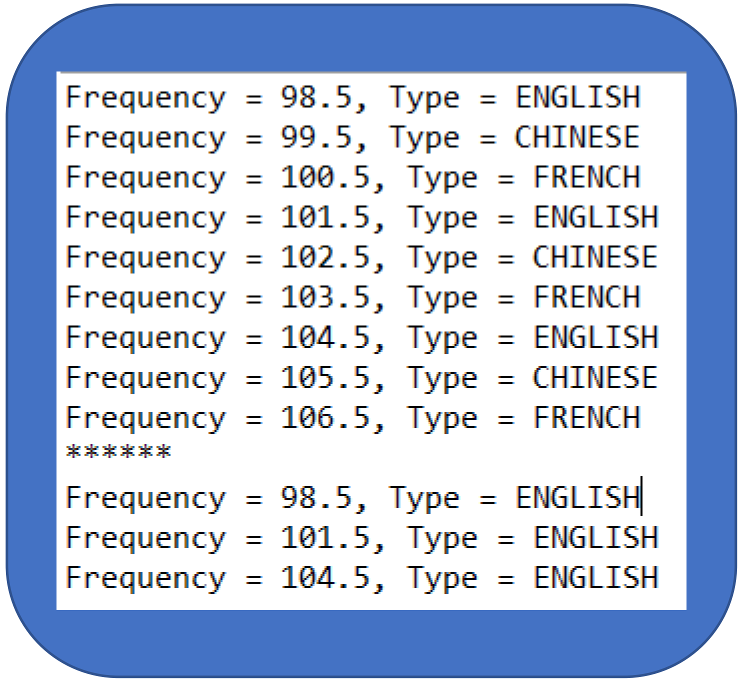
```java
// ConcreteIterator
private class ChannelIteratorImpl implements ChannelIterator {
    private ChannelTypeEnum type;
    private List<Channel> channels;
    private int position;

    public ChannelIteratorImpl(ChannelTypeEnum type,
            List<Channel> channelList) {
        this.type = type;
        this.channels = channelList;
    }

    @Override
    public boolean hasNext() {
        while (position < channels.size()) {
            Channel c = channels.get(position);
            if (c.getType().equals(type) ||
                    type.equals(ChannelTypeEnum.ALL)) {
                return true;
            } else {
                position++;
            }
        }
        return false;
    }
```

```java
    @Override
    public Channel next() {
        Channel c = channels.get(position);
        position++;
        return c;
    }
}
```

代码~Client App



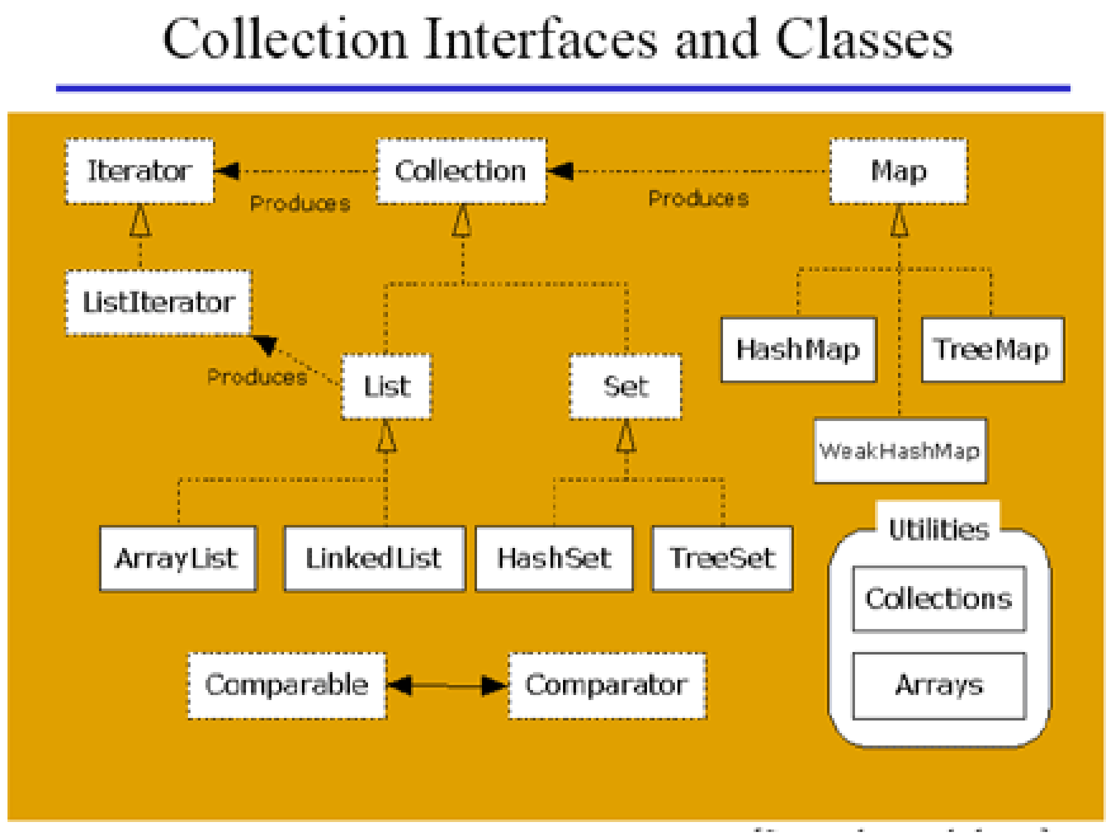
```java
// Client App
public class App {

    public static void main(String[] args) {
        ChannelCollection channels = populateChannels();
        ChannelIterator baseIterator = channels.iterator(ChannelTypeEnum.ALL);
        while (baseIterator.hasNext()) {
            Channel c = baseIterator.next();
            System.out.println(c.toString());
        }
        System.out.println("******");
        // Channel Type Iterator
        ChannelIterator englishIterator = channels.iterator(ChannelTypeEnum.ENGLISH);
        while (englishIterator.hasNext()) {
            Channel c = englishIterator.next();
            System.out.println(c.toString());
        }
    }
 private static ChannelCollection populateChannels() {
     ChannelCollection channels = new ChannelCollectionImpl();
     channels.addChannel(new Channel(98.5, ChannelTypeEnum.ENGLISH));
     channels.addChannel(new Channel(99.5, ChannelTypeEnum.CHINESE));
     channels.addChannel(new Channel(100.5, ChannelTypeEnum.FRENCH));
     channels.addChannel(new Channel(101.5, ChannelTypeEnum.ENGLISH));
     channels.addChannel(new Channel(102.5, ChannelTypeEnum.CHINESE));
     channels.addChannel(new Channel(103.5, ChannelTypeEnum.FRENCH));
     channels.addChannel(new Channel(104.5, ChannelTypeEnum.ENGLISH));
     channels.addChannel(new Channel(105.5, ChannelTypeEnum.CHINESE));
     channels.addChannel(new Channel(106.5, ChannelTypeEnum.FRENCH));
     return channels;
 }
```



```
Frequency = 98.5, Type = ENGLISH
Frequency = 99.5, Type = CHINESE
Frequency = 100.5, Type = FRENCH
Frequency = 101.5, Type = ENGLISH
Frequency = 102.5, Type = CHINESE
Frequency = 103.5, Type = FRENCH
Frequency = 104.5, Type = ENGLISH
Frequency = 105.5, Type = CHINESE
Frequency = 106.5, Type = FRENCH
******
Frequency = 98.5, Type = ENGLISH
Frequency = 101.5, Type = ENGLISH
Frequency = 104.5, Type = ENGLISH
```

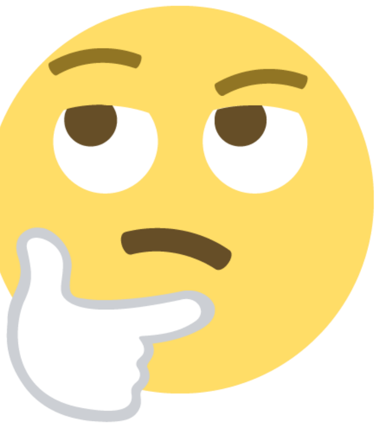
应用

- Java Collection Framework
- java.util.Scanner



课后练习

- 深入调研Java Collection Framework中的Iterator模式



参考

- Iterator Design Pattern in Java
  - https://www.journaldev.com/1716/iterator-design-pattern-java

代码

- https://github.com/spring2go/core-spring-patterns

波波微课
spring2go.com

波波微课
spring2go.com