. Java 8 using Lambdas

Now, let's see how we can do the same in Java 8 by using lambda expression and Stream API, here is my first attempt:

```
Map<String, Book> result  = books.stream()
            .collect(Collectors.toMap(book -> book.getISBN, book -> book));
```

In the above code example, the `stream()` method returns a stream of Book object from the list, and then I have used `collect()` method of Stream class to collect all elements. All the magic of how to collect elements happening in this method.

I have passed the method `Collectors.toMap()`, which means elements will be collected in a Map, where the key will be ISBN code and value will be the object itself. We have used a lambda expression to simplify the code.

3. Using Java 8 method reference

You can further simplify the code in Java 8 by using method reference, as shown below:

```
Map<String, Book> result =  books.stream()
        .collect(Collectors.toMap(Book::getISBN, b -> b));
```

Here we have called the getISBN() method using method reference instead of using a lambda expression.

You can further remove the last remaining lambda expression from this code, where we are passing the object itself by using `Function.identify()` method in Java 8 when the value of the Map is the object itself, as shown below:

```
Map<String, Book> result = choices.stream()
        .collect(Collectors.toMap(Book::getISBN, Function.identity()))
```