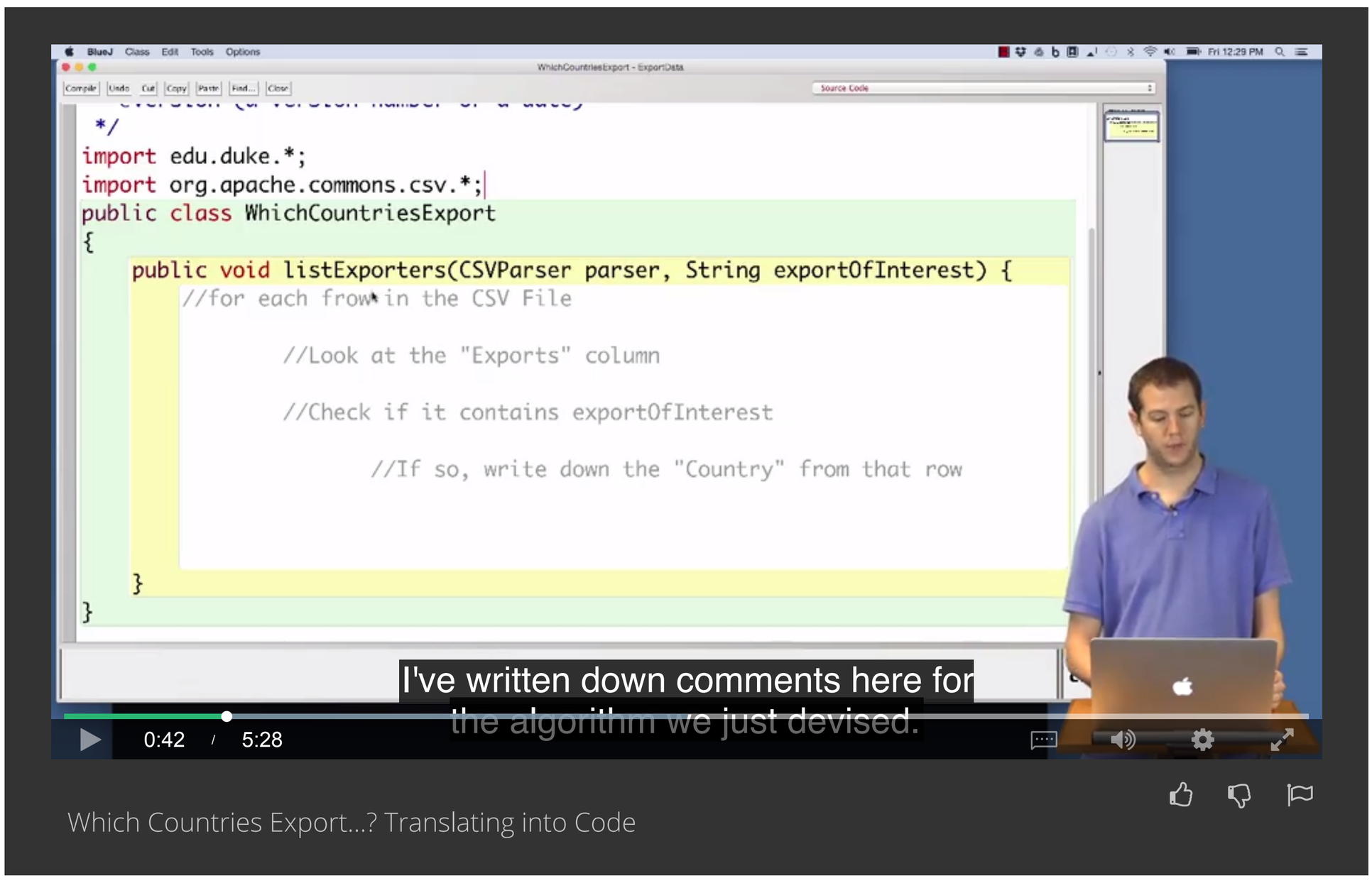

Which Countries Export...? Translating into Code

Have a question? Discuss this lecture in the week forums.

Interactive Transcript

Search Transcript English

0:03
Okay, now that we've developed our algorithm, to figure out which countries export a particular item, it's time to turn that algorithm into code. I'm here in BlueJ, where we've created a class and imported edu.duke.*, which you've used for many things, and org.apache.commens.csv.star, which you've learned recently is what you need for the CSVParser class.

0:28
I've written a method here, listExporters, which takes in a CSVParser, so that's going to be the CSVParser that's already opened for the file you want, and a string for the export of interest, and I've written down comments here for the algorithm we just devised. So the first step in our algorithm, which has a slight typo in it, said for each row in the CSV File, you've already learned that that's for CSVRecord is the type of item that you iterate over in the CSV File. And we'll call it record : parser.

1:09
And I'm going to put curly braces around all of the steps that go inside of there.

1:18
And then, we said look at the export's column. You learned before. I'm just going to call that Exports. This is record.get and the name of the column you want. In this case, Exports.

1:33
Check if it contains exportOfInterest. So now we need to check and see if this string Export contains exportOfInterest. One way we could do that, which you've seen before, is export.indexOf(exportOfInterest) != -1). Now if you were to look in the Java string API you could see that there's a little bit more readable and convenient way to do this, with a method that is just called contains. These two will do the same thing. It's just a little more clear to someone reading the code what you're trying to do, if you write contains, instead of, index of equals negative one.

2:16
Just put that back where it goes.

2:19
And then, we said, if so, write down the country from that row. So I need to first get the country. String country =, and again, I'm going to use record.get("Country") to get the country column from that row. And then, if I want to write something down, I would print it out.

2:43
Okay, and then my curly brace's all match up. I'm just going to hit compile. It says it compiled with no syntax errors. Now, as with other things, making a CSVParser in the BlueJ object workbench is a little complicated, so we're just going to make a method here to help us test this, which is going to be whoExportsCoffee().

3:11
It's going to take no parameters, and it's going to tell us which countries export coffee from a particular data set. So I'm going to make a file resource, which you've seen many times before.

3:24
And as you may recall, if I create a file resource without passing in any parameters, it will pop up a dialog to let me pick what I want to chose it from. And then, I'm going to want to get this CSVParser out of that.

3:42
That file resource has a way to give me a CSVParser, which will parse that file data, and then I want to listExporters(parser, "coffee");. And i'm going to compile this again. It tells me invalid method declaration; return type required. I forgot to tell it what type it returns. In this case, I don't want to return anything since I'm just printing things out, so I'm just going to write void there, and then i'm going to try to compile again. That compiled just fine. I'm gonna come over here, and I'm going to make a new one of my objects, and I'm going to say whoExportsCoffee(). And then, I have this exportdata.csv file, which is going to list a whole bunch of countries.

4:34
Now, we're trying to test out our code, we don't know if this is correct, how would we know this? Well we'd have to go through that entire CSV File and check, did we get everything that exports coffee? That would be really tedious, and is the whole point of writing the program, in the first place. So what we should and can do, instead, is write a much smaller file, here's the one that we used in the slides, and we can test our code on that instead. So if I come back here, and I do whoexports on, this is exportsmall.csv, it says Madagascar and Malawi which we know is the right answer, so that gives us more confidence in our code, and we can be more certain that we got the correct answer on the larger file.

5:21
So there's the code for that, and now you can do similar things yourself.

Downloads

Lecture Video mp4

Subtitles (English) WebVTT

Transcript (English) txt

Would you like to help us translate the transcript and subtitles into additional languages?