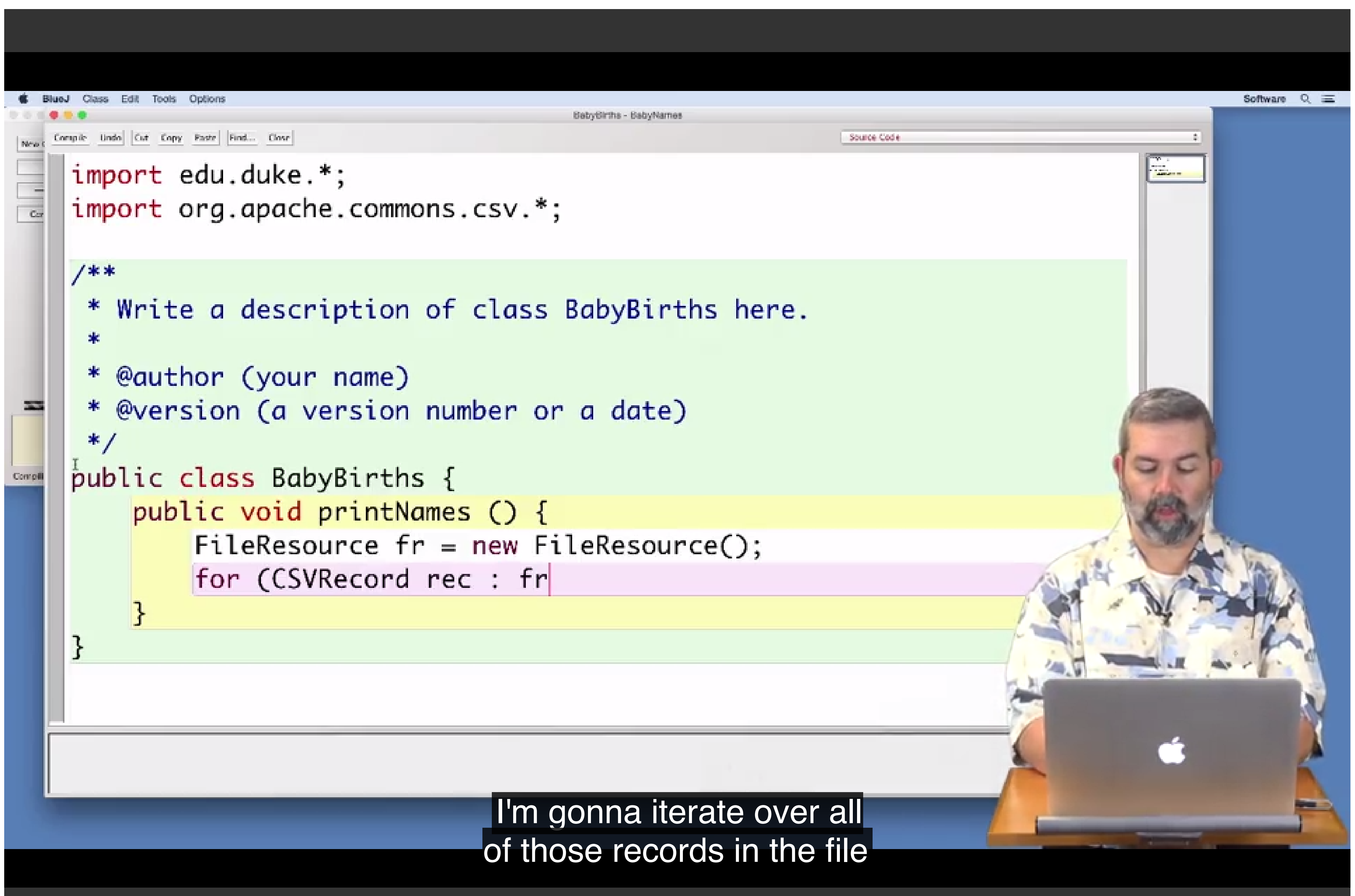
Catalog Search catalog For Enterprise Java

Back to Week 4 Lessons Prev Next

Walkthrough

Baby Names MiniProject: Overview	4 min
Baby Names MiniProject: Data Overview	6 min
Baby Names MiniProject: Total Births	7 min
MiniProject Exercise Guide	10 min
Quiz: Baby Names	14 questions
Extend Your Program	10 min
End of Module Survey	10 min

Batch Grayscale Images



```
import edu.duke.*;
import org.apache.commons.csv.*;

/**
 * Write a description of class BabyBirths here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class BabyBirths {
    public void printNames () {
        FileResource fr = new FileResource();
        for (CSVRecord rec : fr
    }
}
```

I'm gonna iterate over all
of those records in the file

Baby Names MiniProject: Data Overview

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

Interactive Transcript

Search Transcript English

0:03
Now that you've learned about the problem that you're gonna be solving for your mini project, let's take a look at the data files and some code for working with those data files to get us started and working through the problem. For this first example, I'm just going to go ahead and print out some basic information about what's in the data file so that we can get to know it and make sure that we understand what's going on.

0:27
So I'm first gonna make a file resource and I'm gonna choose which file I wanna go ahead and open. And then I'm going to create a CSV record and

0:39
I'm gonna iterate over all of those records in the file by using the CSV parser that we've seen several times. But in a new twist as we saw in our video, we're going to put the data value false when we create the CSV parser and that means that this CSV file does not have a header row. In other words, the very first line of the file is actual data that we're gonna be using instead of the header row.

1:13
For each record that we iterate over, right now we're just going to print out the basic information about it in a nice format so that we can read it and make sure, so I put spaces in between the names. And again, which is different than what we've seen before, we're gonna be accessing our information using numbers instead of names. And that's because again we don't have a header row, so that we're doing it by value where zero is the first one,

1:50
and one is the second one and two is gonna be our third one

1:58
which is all of the fields that we're gonna have in our data.

2:06
So that's it for our first version of this program. We're gonna compile it, and run it on a data file that I've created that is meant to be a very simple example just to get us started. And you can see that Emma is the name right here, Emma's gender is female, and there were 500 girls named Emma born in this example. The second one Olivia, female, 400 born, all the way down to Ava who was the last ranked girl with 100 born.

2:46
Noah, is the first male born, with 100 born, so he was the most babies born, so he's ranked number one in

2:59
terms of male births even though he appears as the sixth name in the file. So it's all females first, and then all males following that according to their rank. So one, two, three, four, five in our example file, one, two, three, four, five in our example file. So we have five girls and five boys.

3:20
You can see the actual data file that I've created here in this spreadsheet where I've loaded the data up. And again, there are five version or five females and five males. And this is what the data file would look like on a small scale so that we have some numbers we can test with. Here's an actual data file where this for the births in the year 2014, and you can see that Emma again is the most popular, but with 20,799 births, instead of the 500 that I made. In this particular example, I've gone ahead and calculated the totals, just to give us a sense of how large this file is. And you can see from the total names, there are 33,044 different names in this file. That means there are 33,044 different lines in this file, because each name occurs on a separate line. Of those, 19,067 are girls' names, so that means the first 19,067 lines in the file are the girls' names, and the 19,068th name in the file is Noah, who is the highest ranked boy for this year. And he would be first of 13,977 boys names that appear throughout the rest of the file. So we're gonna be working for our examples with this small file right here, just so we can get a handle on it cuz 19,000 and 33,000 are big numbers that I can't really work with and test very easily. But we're gonna use these for our testing purposes. So just, again, as a simple little thing to try, we're gonna go ahead and only print out the names if

5:09
the number born is under a certain value. So I'm gonna go ahead and

5:19
Create a integer from the string that we get back for the number born. And so that's gonna be the second piece of information. And then I'm gonna check if that number born is less than or equal to 100, and then print out those names, only those names that are smaller than 100, and so let's see what those are gonna be.

6:05
So now, when I compile it and run it,

6:12
With that same small data,

6:21
I see that I get Ava, who was the last ranked girl, and all of the boys' names that were in there because all of the boys' names had num born less than 100.

Downloads

Lecture Video	mp4
Subtitles (English)	WebVTT
Transcript (English)	txt

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