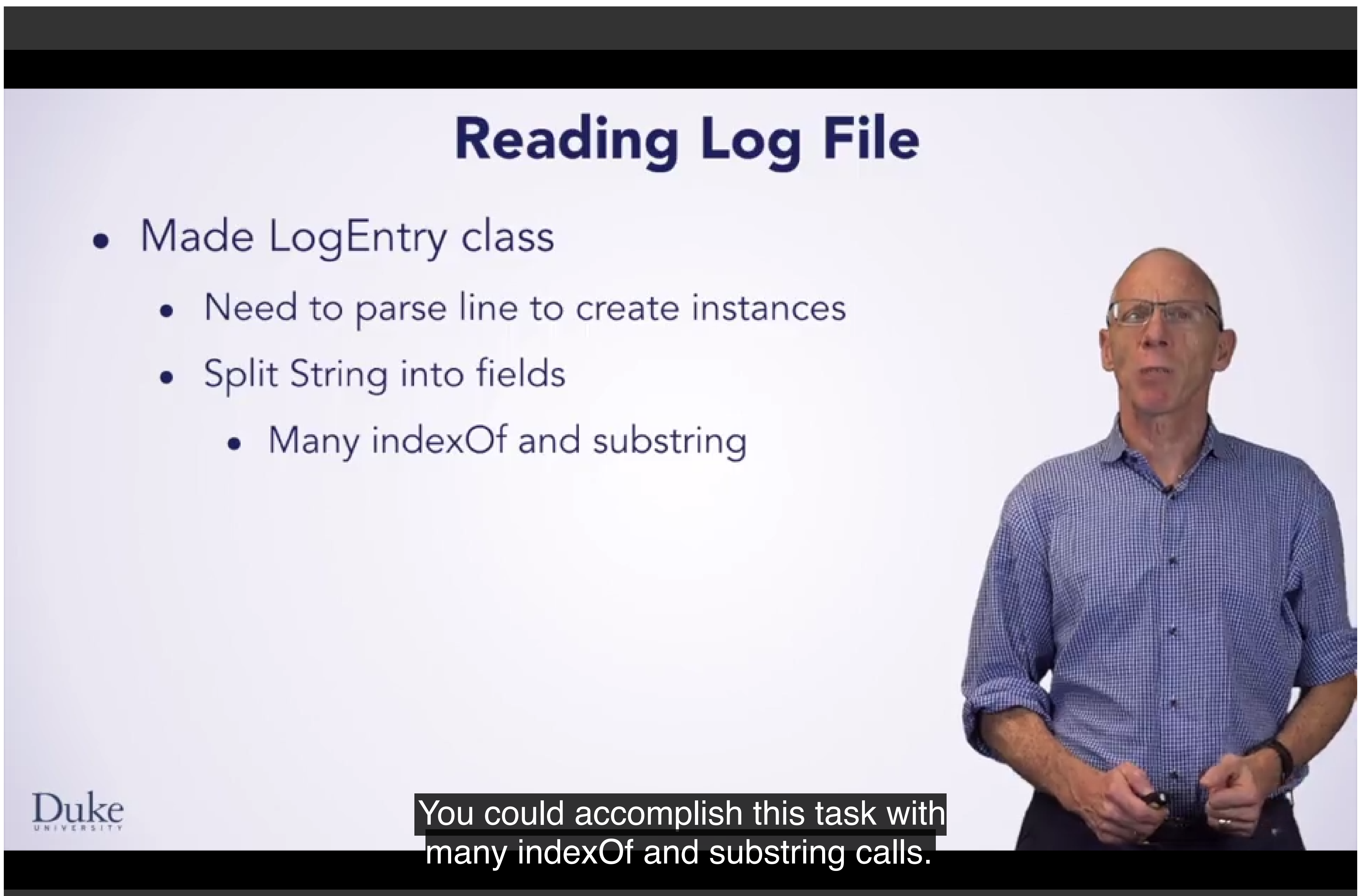
## Reading Log Files

| | |
|---|---|
| Module Learning Outcomes / Resources | 10 min |
| Introduction | 1 min |
| Understanding Log Files | 6 min |
| LogEntry Class with toString | 5 min |
| Parsing Log Files | 3 min |
| Summary | 45 sec |
| Programming Exercise: Reading Log Files | 10 min |
| **Practice Quiz:** Reading Log Files | 3 questions |

## Finding Unique IP Addresses

## Counting Website Visits

## Review



Reading Log File

- Made LogEntry class
  - Need to parse line to create instances
  - Split String into fields
    - Many indexOf and substring

Duke
UNIVERSITY
You could accomplish this task with many indexOf and substring calls.

Parsing Log Files

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

## Interactive Transcript

Search Transcript English

0:03
Hi, now that you've made your LogEntry class, you need to parse the lines of the web server log to be able to create instances of the LogEntry class. You'll do this by splitting the string into the appropriate fields to pass values to the constructor for the LogEntry class. You could accomplish this task with many indexOf and substring calls. Although this task is not algorithmically hard, the code for it is very cumbersome. For example, for the time portion of this entry, you would need to turn the string in to a Date object. The built in Java class from the Java.util package which represents a date and time. Even though both the date and time class, as well as methods which parse strings, are part of Java, the interface to the Date class is complex, especially since the date format in the server logs is not the default format in Java. For these reasons, we've provided code for you which will take a string from the web server logs, parse it into appropriate fields, and return a log entry record. To use this call WebLogParser.parseEntry and pass the string you want to parse. The method returns a LogEntry object.

1:16
With that in mind, its time for you to starting the write the LogAnalyzer class. For right now, you're gonna write code in the constructor to initialize the object, and then write the read file method. In later lessons, you'll write additional methods that will perform that actually analysis of the log file that you've read in.

1:36
The first thing you would do to fill in the code for the constructor. The constructor should initialize that the record fields to an empty ArrayList. You've created ArrayLists in the past so what you need to accomplish this task should be familiar. The second thing you should do is fill in code for the readFile method.

1:55
This method will determine the file name to read from and then add log entries to the records field to reflect the information from the file you opened. To accomplish this task, you will want to make a FileResource for the requested file. You will then want to iterate over the FileResource's lines. And for each line, you will use the WebLogParser.parseEntry method to convert the line of text into a LogEntry. Then you'll add that LogEntry to the records field, which as you may recall, is an ArrayList.

2:30
When you've written a constructor and the readFile method, you'll want to test out your code. We've provided a convenient method called printAll, which will print all the log entries you've stored in the instance variable records. Remember the .toString method that we taught about? System.out.println will make use of that .toString method to represent the log entry as a string. Once all this works, it will be time to start analyzing the data you've read in. Happy coding.

## Downloads

| | |
|---|---|
| **Lecture Video** | mp4 |
| **Subtitles (English)** | WebVTT |
| **Transcript (English)** | txt |

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