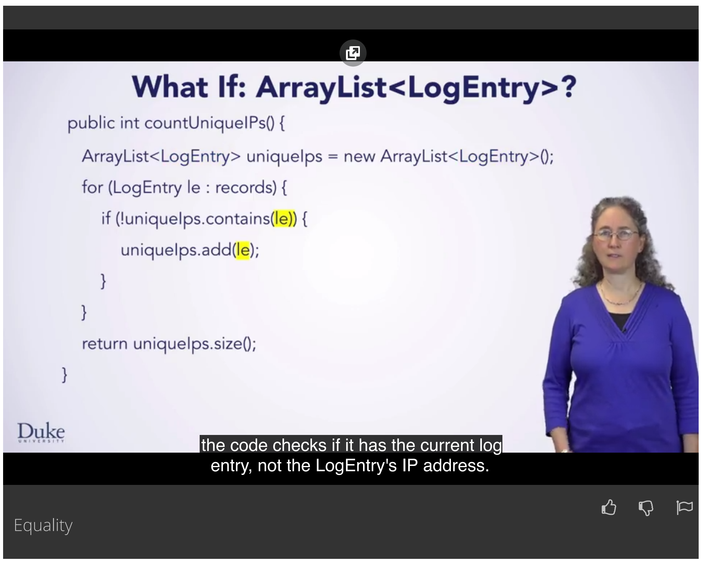

Equality

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

Interactive Transcript

Search Transcript

English

0:03
Now that you have written code to find the unique IP addresses in a web server log, let us take a brief look at something very close that would not have worked.

0:13
Here is the code that you just wrote for the unique IP addresses problem. Remember that you have an array list of strings and that you put the string for each log entries IP address into that array list.

0:26
But what if you had written this code instead? This code is the same as what you wrote, but the array list holds log entries and the code checks if it has the current log entry, not the LogEntry's IP address. Likewise, it adds the entire LogEntry object, if it was not already there. If you were to run this code, it would give you the wrong answer. In fact, it would tell you how many total LogEntry's there are, not how many unique IP addresses there are in the log file. Why is that?

1:00
To understand this problem, think for a moment about how contains would work. In particular, how does contains know if two objects are the same or different? Contains is going to check if they are equal. What exactly do we mean by equal?

1:16
Java has two different notions of equality. To illustrate this, consider the situation in which you have three string variables, a,b, and c, which refer to two different string objects. a and b refer to the exact same string object, so they are clearly equal. a and c however refer to different string objects with the same logical contents.

1:38
On the one hand you could say they are equal, because they talk about strings that mean the same thing. On the other you could say they are not, because they are talking about different objects. These are the two different notions of equality that exist in Java.

1:54
The notion of equality meaning the exact same object is what you get when you write, = =. If you write, a = = b, then Java checks if a and b refer to the exact same object. Since they do, this expression evaluates to true. However, if you write a == c, then Java again checks if a and c refer to the exact same object. But because they do not, this expression evaluates to false. The other notion of equality, do they mean the same thing, is done with a .equals method. If you wrote a.equals(c), then Java would call the .equals method in the string class, which checks if the two strings have the same sequence of characters. Because these two strings have the same sequence of characters, a.equals(c), would evaluate to true? So how does Java know whether two objects have the same logical meaning? Each class defines .equals to specify what it means for objects of that type to be the same.

2:56
If you do not write one explicitly, the default behavior would be to have the .equals method check if the two objects are = = to each other? That is if they are the exact same objects.

3:09
So know that you know about equals equals and .equals should you write a .equals method for log entry?

3:17
Well the first thing you should do is think about when 2 log entries are logically the same. What about if they have the same IP address? Well that would fix the broken code for this particular problem, it's not a good approach in general. It does not actually match with the notion of two log entries meaning the same thing.

3:36
Two different requests are not the same, even if they came from the same computer. So what if you checked more information? What if you checked for the same IP address and the same request string?

3:47
Even this would not really mean the two log entries are the same, as the same computer could ask for the same page many times.

3:55
For log entries it makes sense to just say that they are logically the same, only if they are in fact the exact same object.

4:03
Because the behavior you want is the default for .equals, you do not need to explicitly write it.

4:10
Since you do not need to write a .equals method for this class, we're not going to delve into how to do it yet.

4:16
Fully understanding what goes into a .equals method requires a little bit of knowledge that you will not get until the principals of software design course.

4:26
However, now that you understand the ideas of equality and that the contains method checks if two objects are the same you can understand why this code did not work and this code where you used the IP addresses did work. Thanks.

Downloads

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

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