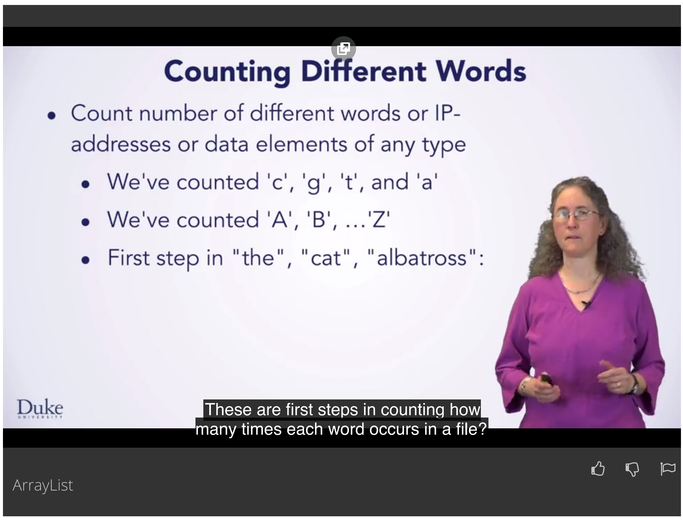
Telling a Random Story

Using and Improving GladLibs

Review



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

Interactive Transcript

Search Transcript English

0:03
Welcome back. In this lesson, you'll learn about a new class. A very important part of Java that combines essential features of the storage resource class, and an array. In fact, this new class, the ArrayList is the foundation of how the stored resource class is implemented. To motivate the problem, think about counting how many different words there are in a file or a webpage. But the same problem is encountered in finding the number of unique IP addresses to visit a website in a day, a key part of charging a company for online advertising. You've seen how to count the number of each type of nucleotide in digitized DNA. And you used an array to count the number of occurrences of each alphabetic character in cracking a Caesar cipher. These are first steps in counting how many times each word occurs in a file? How many times the occurs or cat or the word albatross? One important part of solving that problem is finding how many different words there are, so that the counts as one word, rather than 573, if it occurs that many times in document. If the image below In the image below, there are only three different digits, four, six, and seven, even though there are hundreds of digits shown. We'll look at StorageResource as a way of solving this problem. First, we'll count the total number of words in a file or web page. The class StorageResource makes this easy. To count the words in a file or web page you'll iterate over either a file resource or URL resource. As you can see in the highlighted code, iterating over a file or webpage using these resources uses nearly identical code. And here you see that simply calling the dot add method adds each word to the StorageResource instance variable my words. When done, the .size method will provide the total number of words read. The method .getCount returns this number when called again, using the instance variable myWords, which was initialized in the constructor and added to in the method readWords. As you'll see, it's easy to use StorageResources to count the number of different words. Not simply the total number of words.

2:15
This code can be easily modified to find the number of unique or different words in a file or webpage.

2:22
The field, myWords, whose type is storage resource, can store all the words as you've just seen and as shown in the code here.

2:31
The .add method adds every string read to myWords. But it's simple to guard the call to .add with code that only calls .add when the word is seen for the first time. When it hasn't been stored in myWords yet. The .contains method returns a Boolean value. And the code here, this value is used to ensure that the .add method is called only when the storage research object my words does not contain a word.

2:59
However, the storage research class is not a good choice for choosing elements at random. The key part of the story telling code glad lids we're working on.

3:09
To choose an element at random, we must use the iterable interface that StorageResource provides. This means we use a loop to access every element in the StorageResource object myWords. In the loop here, we'd really like to iterate as many times as the values stored in variable choice, because we want to use a random element of the StorageResource. The code here returns a random string when the value of counter reaches zero. As you can see in the code, the value of choice must reach zero because it starts at a value less than the size of my words and it's decremented by one each time through the loop. However, the Java compiler analyzes the loop with an IF statement, and doesn't know that it's possible. The if statement must be true at some point. The compiler indicates it's an error to be missing a return statement, after the loop even though that part of the code is never reached. It would be simpler to code, and much, much faster to use a string array to get a random element, as shown in the code here. We simply generate a random manager and use that as an index into the array.

4:20
Unfortunately, we must specify the capacity of an array when we declare it. Arrays do not grow in the way that a storage resource object grows.

4:29
The class array list provides a solution and combines the best feature of storage resource and arrays. The class ArrayList is from the java.util package. The same package that contains the random class we've used.

4:43
An ArrayList expands its capacity as needed when it's .add method is called, just like a StorageResource object. An ArrayList also provides access via indexing, so that the zeroth or 101st element can be accessed without iterating over all elements just like an array.

5:01
The StorageResource class uses an ArrayList internally. In fact, it's simply a little easier to use than an ArrayList but as you become more experience, you're now able to use ArrayList which stores any kind of object, not just strings. The basic syntax of the ArrayList class is shown here. But you'll see it used in a coding example in the next lesson.

5:21
To declare an ArrayList variable, you must include the type of object stored in the ArrayList using the angle brackets as shown here.

5:29
Like any object, creating one requires calling new and providing the class name as a constructor to create and initialize the object that's shown here, and assigned to the variable words.

5:42
Then you can call the .add method to add strings to the ArrayList object. And you can call the .get method to access a particular element via indexing. Just like the bracket notation used with an array. But with an ArrayList, you'll use the .get method. The .set method can change or set the element at a particular index in the array list object.

6:05
Here the first or zeroth indexed element is set to the string goodbye.

6:11
We'll see more examples as we go through a coding example with ArrayList. The ArrayList class is more powerful than StorageResource and can sort any kind of object. It will be an essential class in solving many problems with Java. Thank you.

Downloads

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

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