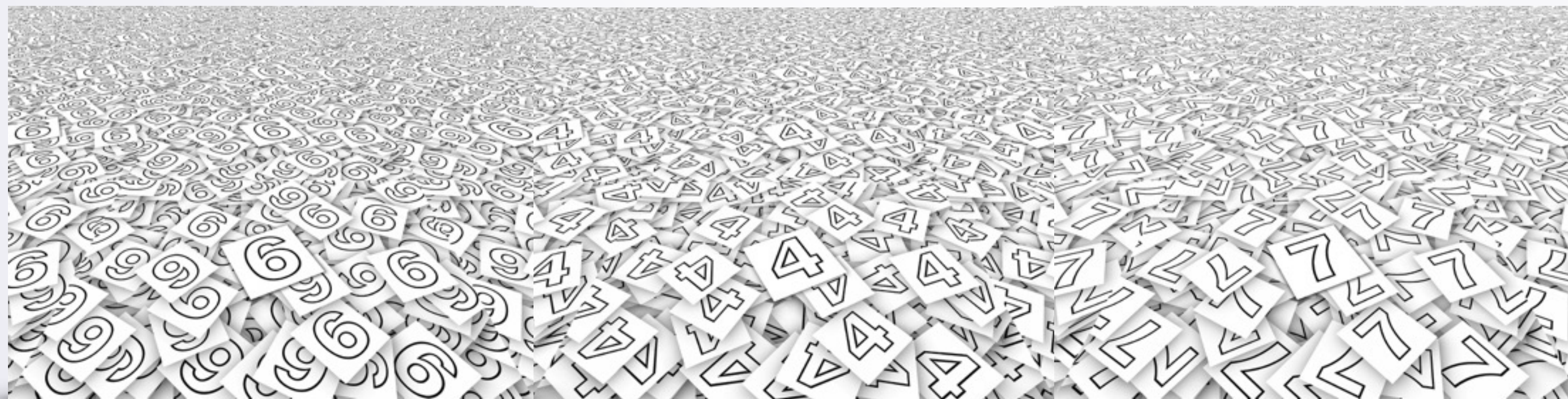


Telling a Random Story

ArrayList

Counting Different Words

- Count number of different words or IP-addresses or data elements of any type
 - We've counted 'c', 'g', 't', and 'a'
 - We've counted 'A', 'B', ... 'Z'
 - First step in "the", "cat", "albatross":
 - Count number of different words



Using StorageResource

- Using StorageResource makes it easy

```
public class CountWords {  
    StorageResource myWords;  
  
    public CountWords() {  
        myWords = new StorageResource();  
    }  
  
    public int getCount(){  
        return myWords.size();  
    }  
  
    public void readWords(String source){  
        myWords.clear();  
        if (source.startsWith("http")){  
            URLResource resource = new URLResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
        else {  
            FileResource resource = new FileResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
    }  
}
```

Using StorageResource

- Using StorageResource makes it easy
 - To count all words in a file or URL

```
public class CountWords {  
    StorageResource myWords;  
  
    public CountWords() {  
        myWords = new StorageResource();  
    }  
  
    public int getCount(){  
        return myWords.size();  
    }  
  
    public void readWords(String source){  
        myWords.clear();  
        for(String word : resource.words()){  
            myWords.add(word.toLowerCase());  
        }  
    }  
    else {  
        FileResource resource = new FileResource(source);  
        for(String word : resource.words()){  
            myWords.add(word.toLowerCase());  
        }  
    }  
}
```

Using StorageResource

- Using StorageResource makes it easy
 - To count all words in a file or URL
 - Add each to StorageResource

```
public class CountWords {  
    StorageResource myWords;  
  
    public CountWords() {  
        myWords = new StorageResource();  
    }  
  
    public int getCount(){  
        return myWords.size();  
    }  
  
    public void readWords(String source){  
        myWords.clear();  
        if (source.startsWith("http")){  
            URLResource resource = new URLResource(source);  
            myWords.add(word.toLowerCase());  
            myWords.add(word.toLowerCase());  
        }  
        else {  
            FileResource resource = new FileResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
            myWords.add(word.toLowerCase());  
        }  
    }  
}
```

Using StorageResource

- Using StorageResource makes it easy
 - To count all words in a file or URL
 - Add each to StorageResource
 - Use .size()

```
public class CountWords {  
    StorageResource myWords;  
  
    public CountWords() {  
        myWords = new StorageResource();  
    }  
  
    public int getCount(){  
        return myWords.size();  
    }  
  
    public void readWords(String source){  
        myWords.clear();  
        if (source.startsWith("http")){  
            URLResource resource = new URLResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
        else {  
            FileResource resource = new FileResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
    }  
}
```


Using StorageResource

- Using StorageResource makes it easy
 - To count all words in a file or URL
 - Add each to StorageResource
 - Use .size()
 - Different words?

```
public class CountWords {  
    StorageResource myWords;  
  
    public CountWords() {  
        myWords = new StorageResource();  
    }  
  
    public int getCount(){  
        return myWords.size();  
    }  
  
    public void readWords(String source){  
        myWords.clear();  
        if (source.startsWith("http")){  
            URLResource resource = new URLResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
        else {  
            FileResource resource = new FileResource(source);  
            for(String word : resource.words()){  
                myWords.add(word.toLowerCase());  
            }  
        }  
    }  
}
```

Modifying Code for Unique Words

- Field: `StorageResource myWords`
 - Store all words read from a file

```
FileResource resource = new FileResource(source);  
for(String word : resource.words()){  
    myWords.add(word.toLowerCase());  
}
```


Modifying Code for Unique Words

- Field: `StorageResource myWords`
 - Store all words read from a file
 - only unique/different words

```
FileResource resource = new FileResource(source);  
for(String word : resource.words()){  
    myWords.add(word.toLowerCase());  
}
```

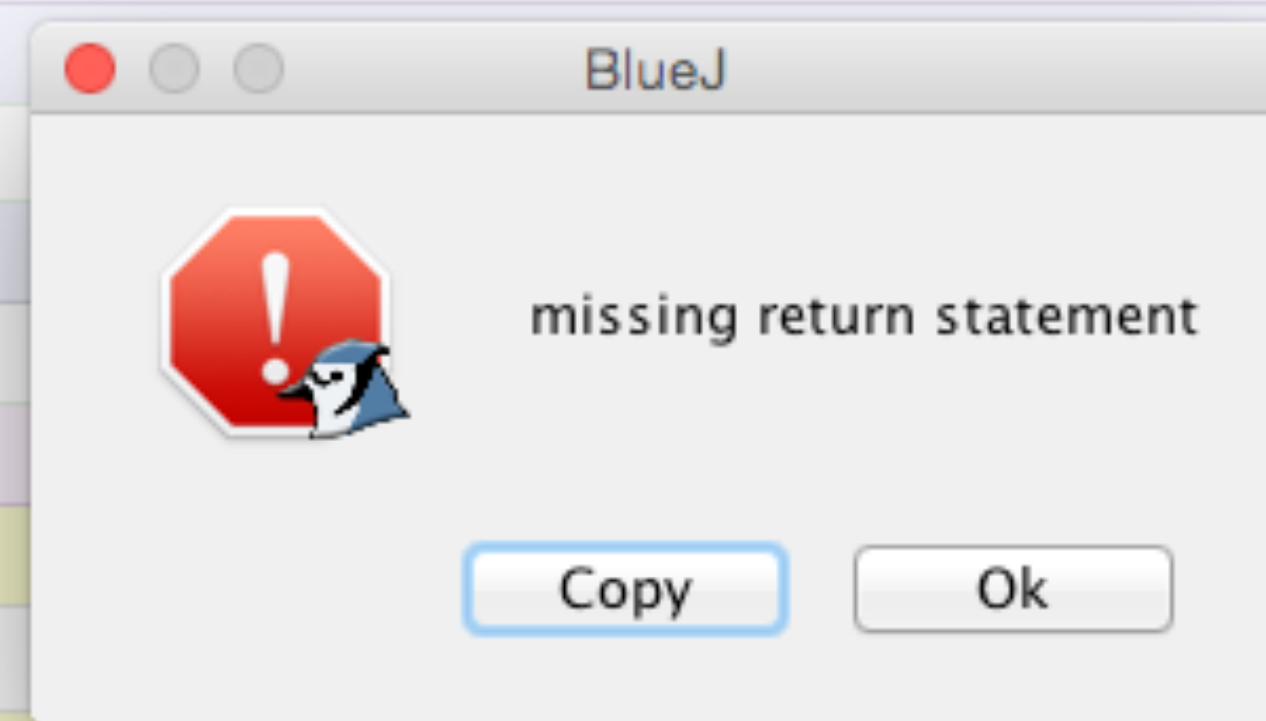
```
FileResource resource = new FileResource(source);  
for(String word : resource.words()){  
    word = word.toLowerCase();  
    if (! myWords.contains(word)){  
        myWords.add(word);  
    }  
}
```

Random Choice from StorageResource

- StorageResource accessed as iterable
 - Must use for loop to get at all elements
 - Even if we stop early, coding issues

```
public String getRandomWord(){
    Random rand = new Random();
    int choice = rand.nextInt(myWords.size());
    for(String s : myWords.data()){
        if (choice == 0) {
            return s;
        }
        choice = choice - 1;
    }
}

public void readWords(String source){
```



Random Choice from StorageResource

- StorageResource accessed as iterable
 - Must use for loop to get at all elements
 - Even if we stop early, coding issues
- Would be faster and simpler with String[]
 - But don't know capacity before reading!

```
public String getRandomWord(String[] words) {  
    Random rand = new Random();  
    int index = rand.nextInt(words.length);  
    return words[index];  
}
```


ArrayList as a Solution

- Class **ArrayList** in package `java.util`
 - Expands as needed using `.add` method
 - Provides access via index to any element in list
 - Essential in implementing `StorageResource`!
- Basic syntax, we'll see usage in code

```
ArrayList<String> words = new ArrayList<String>();  
words.add("hello");  
words.add("world");  
String s = words.get(1);  
words.set(0, "goodbye");
```

ArrayList as a Solution

- Class **ArrayList** in package `java.util`
 - Expands as needed using `.add` method
 - Provides access via index to any element in list
 - Essential in implementing `StorageResource`!
- Basic syntax, we'll see usage in code

```
ArrayList<String> words = new ArrayList<String>();  
words.add("hello");  
words.add("world");  
String s = words.get(1);  
words.set(0, "goodbye");
```

ArrayList as a Solution

- Class **ArrayList** in package `java.util`
 - Expands as needed using `.add` method
 - Provides access via index to any element in list
 - Essential in implementing `StorageResource`!
- Basic syntax, we'll see usage in code

```
ArrayList<String> words = new ArrayList<String>();  
words.add("hello");  
words.add("world");  
String s = words.get(1);  
words.set(0, "goodbye");
```


ArrayList as a Solution

- Class **ArrayList** in package `java.util`
 - Expands as needed using `.add` method
 - Provides access via index to any element in list
 - Essential in implementing `StorageResource`!
- Basic syntax, we'll see usage in code

```
ArrayList<String> words = new ArrayList<String>();  
words.add("hello");  
words.add("world");  
String s = words.get(1);  
words.set(0, "goodbye");
```

ArrayList as a Solution

- Class **ArrayList** in package `java.util`
 - Expands as needed using `.add` method
 - Provides access via index to any element in list
 - Essential in implementing `StorageResource`!
- Basic syntax, we'll see usage in code

```
ArrayList<String> words = new ArrayList<String>();  
words.add("hello");  
words.add("world");  
String s = words.get(1);  
words.set(0, "goodbye");
```