

■ Reverse Words in a String

Flip the word order while cleaning up extra spaces

● String

■ MEDIUM

■ What You Will Learn

- How to split a string and handle multiple spaces
- The two-pointer reverse trick on a word array
- Java's built-in String tricks (trim, split regex)
- Why 'reverse all, then reverse each word' is elegant

■ Intuition First

Input: ' the sky is blue ' → Output: 'blue is sky the'. You want words in reverse order, but no leading/trailing/extra spaces. Think of cutting the sentence into word cards, flipping the pile, then reading them out.

■ Key Insight

The hardest part is the spaces — multiple spaces between words, spaces at start/end. Java's `split("\\s+")` handles ALL of these in one shot. After splitting, just reverse the word array.

■ Visual Explanation

```
Input: ' the sky is blue '
      ↓ trim + split on spaces
Words: ['the', 'sky', 'is', 'blue']
      ↓ reverse array
Words: ['blue', 'is', 'sky', 'the']
      ↓ join with single space
Output: 'blue is sky the'

Two-pointer reverse on array:
[the | sky | is | blue]
  L           R   → swap → [blue | sky | is | the]
[blue | sky | is | the]
      L   R       → swap → [blue | is | sky | the]
Done! L >= R
```

■ Java Code

```
1  class Solution {
2      public String reverseWords(String s) {
3          // trim edges, split on any whitespace sequence
```

```

4   String[] words = s.trim().split("\\s+");
5
6   // Two-pointer reverse
7   int left = 0, right = words.length - 1;
8   while (left < right) {
9       String temp = words[left];
10      words[left] = words[right];
11      words[right] = temp;
12      left++; right--;
13  }
14  return String.join(" ", words);
15  }
16  }

```

■ Complexity

Type	Value	Meaning
Time	$O(n)$	One pass to split, one to reverse, one to join
Space	$O(n)$	The words array holds all characters

■ Common Mistakes

■ Mistake

Using `split(' ')` with a single space. This fails on multiple spaces — use `split("\\s+")` which handles any whitespace sequence.

■ Pro Tips

■ In-place approach

For $O(1)$ space: reverse entire string char-by-char, then reverse each word individually. Interviews love this approach!

■ Quick Revision

✓	<code>trim()</code> removes leading/trailing spaces
✓	<code>split("\\s+")</code> handles multiple spaces between words
✓	Reverse the word array with two pointers
✓	<code>String.join(' ', words)</code> reassembles with single spaces
✓	Time $O(n)$, Space $O(n)$