👍 244   👎 130   ♡   ▼

804. Unique Morse Code Words

📋 Description (/problems/unique-morse-code-words/description/)   💡 Hints (/problems/unique-morse-code-words/hints/)   📄 Submit

International Morse Code defines a standard encoding where each letter is mapped to a series of dots and dashes, as follows: `"a"` maps to `".-"` , `"b"` maps to `"-..."` , `"c"` maps to `"-.-."` , and so on.

For convenience, the full table for the 26 letters of the English alphabet is given below:

```
[".-","-...","-.-.","-..",".","..-.","--.","....","..",".---","-.-",".-..","--","-.","-
```

Now, given a list of words, each word can be written as a concatenation of the Morse code of each letter. For example, "cab" can be written as "-.-.-....-", (which is the concatenation "-.-." + "-..." + ".-"). We'll call such a concatenation, the transformation of a word.

Return the number of different transformations among all words we have.

```
Example:
Input: words = ["gin", "zen", "gig", "msg"]
Output: 2
Explanation:
The transformation of each word is:
"gin" -> "--...-."
"zen" -> "--...-."
"gig" -> "--...--."
"msg" -> "--...--."


There are 2 different transformations, "--...-." and "--...--.".
```

Note:

- The length of `words` will be at most `100` .
- Each `words[i]` will have length in range `[1, 12]` .
- `words[i]` will only consist of lowercase letters.

Seen this question in a real interview before? Yes No   ✕

Companies ⓘ ▲

Google   4   (/company/google)

Related Topics ▲

String (/tag/string)

Java ▼   〈/〉   ↻   >_   ⚙   ▢ Notes

```java
class Solution {
    public int uniqueMorseRepresentations(String[] words) {
        Set<String> unique = new HashSet<>();
        for (String s : words) {
            unique.add(transform(s));
        }
        return unique.size();
    }

    private static final String[] MAPPING = new String[] {".-
","-...","-.-.","-..",".","..-.","--.","....","..",".---","-.-",".-..","--
","-.","---",".--.","--.-",".-.","...","-","..-","...-",".--","-..-","-.--
","--.."};

    private String transform(String s) {
        StringBuilder sb = new StringBuilder();
        for (char ch : s.toCharArray()) {
            sb.append(MAPPING[ch - 'a']);
        }
        return sb.toString();
    }
}
```

☐ **Custom Testcase** (**Contribute** ❶)

▶ Run   ☁ Submit