Daily Coding Problem #176

Problem

This problem was asked by Bloomberg.

Determine whether there exists a one-to-one character mapping from one string `s1` to another `s2`.

For example, given `s1 = abc` and `s2 = bcd`, return `true` since we can map `a` to `b`, `b` to `c`, and `c` to `d`.

Given `s1 = foo` and `s2 = bar`, return `false` since the `o` cannot map to two characters.

Solution

We can solve this question by creating a mapping and try to fill it out as we zip along both strings. Let's call the characters at each index `i` `char1` and `char2` for `s1` and `s2` respectively. Then we have to deal with the following cases:

- If the lengths of the strings are different, then return false -- a mappig can't exist.
- If `char1` doesn't exist in the mapping, then create it and set its value to `char2`.
- If `char1` exists in the mapping and if its value is `char2` then continue.
- If `char1` exists in the mapping but its value is not `char2` then we have a conflict, so we can't create a one-to-one mapping, so return `false`.

```
def mapping_exists(s1, s2):
    if len(s1) != len(s2):
        return False

    mapping = {}
    for char1, char2 in zip(s1, s2):
        if char1 not in mapping:
            mapping[char1] = char2
        elif mapping[char1] != char2:
            return False

    return True
```

This takes O(n) time and space.