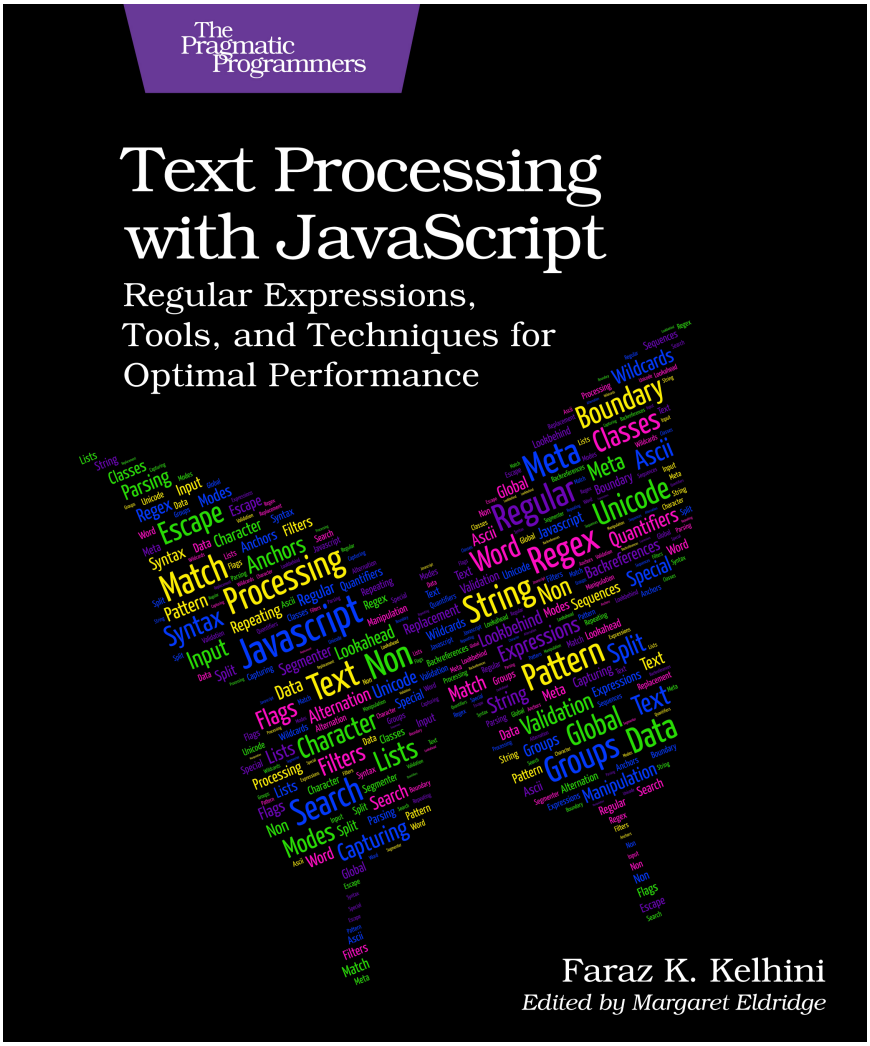

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Recipe 2

Checking a String for Specific Words with includes()

Task

Suppose you're building an online bakeshop and want to filter messages so they can be routed to the correct baker. You need to check the strings in incoming emails to account for different spellings of words like "doughnut" versus "donut." You can't use the includes() method alone because it allows you to look for only a single word.

Solution

Put the words you want to search for in an array. Then create a function that accepts two arguments: a string to search and an array of words. Inside the function, search for each word in the string and return true if at least one search is successful:

part_1/checking_specific_words/includes_ex1.js

```
const msg = "I'd like to order two donuts";
const words = ["doughnut", "donut"];

function hasSomeWords(str, arr) {
  return arr.some(el => str.includes(el));
}

hasSomeWords(msg, words);    // → true
```

The some() method returns true if at least one element in the array passes the test implemented by the given function. In this case, that means includes() first searches for "doughnut." Since there's no such a word in the string, the method returns false. The second time includes() searches for "donut," and this time it returns true. So, the return value of some() will be true.

Discussion

ECMAScript added includes() to the language in ES2015 to enable developers to easily determine whether a string contains another string. The second argument of includes() is optional and lets you specify the position at which to begin searching. For example:

part_1/checking_specific_words/includes_ex2.js

```
const quote = "Sachertorte is a torte of Austrian origin.";
```

```
quote.includes("Sachertorte", 15);    // → false
```

This code starts the search at index 15. Because no word matches “Sachertorte” from index 15 onwards, the return value is false.

Remember, includes() is case sensitive. If you search for “Sachertorte” in a string containing “SacherTorte,” the result is false:

part_1/checking_specific_words/includes_ex3.js

```
const quote = "I'd like to order a SacherTorte.";
const word = "Sachertorte";

quote.includes(word);    // → false
```

Some desserts could have internal capitalization because they are made up of two words such as Dobostorta/DobosTorta, Leibnizkeks/LeibnizKeks, and SacherTorte/Sachertorte. So, in most cases, you want to perform a case-insensitive search by converting both the string and the keyword to lowercase, like this:

part_1/checking_specific_words/includes_ex4.js

```
const quote = "I'd like to order a SacherTorte.".toLowerCase();
const word = "Sachertorte".toLowerCase();

quote.includes(word);    // → true
```

But what if you want to check if a string contains multiple words simultaneously? In that case, you should use the every() method. every() is similar to some() in that it executes a function for each element of an array. But it returns true only if every item in the array passes the test. Here’s an example:

part_1/checking_specific_words/includes_ex5.js

```
const msg = "1 sachertorte, 3 pretzels, and 2 donuts please.";
const wordsArr1 = ["sachertorte", "donut"];
const wordsArr2 = ["sachertorte", "sourdough"];

function hasEveryWord(str, arr) {
  return arr.every(el => str.includes(el));
}

hasEveryWord(msg, wordsArr1);    // → true
hasEveryWord(msg, wordsArr2);    // → false
```

Here, “sachertorte” and “donut” pass the test because they both exist in the string, but that’s not the case for “sachertorte” and “sourdough.”

Although includes() is designed to search for only a single word, with a little effort, you can take advantage of it to search for more words. But be careful when looking for words that also have a compound form.

If you search for “cake” in “I’d like to order two pancakes,” includes() returns true. If you don’t want that to happen, you should use a regex token known as a word boundary. See Recipe 25, Looking For Whole Words Only with the Word Boundary (\b), on page ?.