UDC 004.4'22

CONVERTING SPEECH TO TEXT USING JAVASCRIPT

M. LAPKOUSKI, D. PIATKIN
Polotsk State University, Belarus

This article is about the process of working with the Web Speech API. This is a very powerful browser interface that allows to record human speech and convert it to text. We will also consider the way to use it for the opposite procedure of reading lines with a human voice.

Convert speech to text. The Web Speech API is actually divided into two completely independent interfaces: SpeechRecognition for understanding human voice and turning it into text, and SpeechSynthesis for reading lines aloud with a computer-generated voice.

The speech recognition API is surprisingly accurate for the free browser feature. It recognized correctly almost all my conversations and knew which words are combined into meaningful phrases. It also allows you to dictate special characters such as breakpoints, question marks, and newlines. An example is shown in Listing 1 [1].

The first thing to do is to check if the user has access to the API and show the corresponding error message. Unfortunately, the speech-to-text conversion API is only supported in Chrome and Firefox (with a checkmark), so many users are likely to see this message.

Listing 1 - checking if the user has access to the API

```
try {
  var SpeechRecognition = window.SpeechRecognition || window.webkitSpeechRecognition;
  var recognition = new SpeechRecognition();
}
catch(e) {
  console.error(e);
  $('.no-browser-support').show();
  $('.app').hide();
}
```

Recognition variable will give us access to all API methods and properties. There are various options available, but we will only set the recognition.continuous parameter. This will allow users to speak with longer pauses between words and phrases.

Before we can use voice recognition, we also need to set up several event handlers. Most of them just listen to changes in recognition status. An example is shown in Listing 2.

Listing 2 - setting up event handlers before using voice recognition

```
try {
  var SpeechRecognition = window.SpeechRecognition || window.webkitSpeechRecognition;
  var recognition = new SpeechRecognition();
}
catch(e) {
  console.error(e);
  $('.no-browser-support').show();
  $('.app').hide();
}
```

However, there is a special onresult event, which is very important. It is executed every time the user says a word or several words in quick succession, giving us access to a text transcription of what was said.

When we write something using the onresult handler, we save it in a global variable and display it in the text area. An example is shown in Listing 3.

Listing 3 - recognition.onresult event handler

```
recognition.onresult = function(event) {

  // event is a SpeechRecognitionEvent object.
  // It stores the entire recorded dialog
```

```
  // We only need the current dialog / phrase
 var current = event.resultIndex;

  // Get the text of what was said.
  var transcript = event.results[current][0].transcript;

  // Add text to the content of our text document.
  noteContent += transcript;
  noteTextarea.val(noteContent);
 }
```

The code above is a bit simplified. There is a very strange error on Android devices, due to which everything repeats twice. There is no official solution, but I managed to solve the problem without any obvious side effects. With this error in mind, the code looks like this [2]:

Listing 4 - a simplified code from the previous listing that solves the problem of repeating the output text twice on Android devices

```
var mobileRepeatBug = (current == 1 && transcript == event.results[0][0].transcript);

if(!mobileRepeatBug) {
  noteContent += transcript;
  noteTextarea.val(noteContent);
}
```

Everything is set up, now you can start using the browser voice recognition function. To start with, you just need to call the start () method:

Listing 5 - calling the start () method to start voice recognition

```
$('#start-record-btn').on('click', function(e) {
  recognition.start();
});
```

In this case, the browser will ask users to give permission. If permission is given, the device microphone will be activated.

The browser will listen for some time, and each recognized phrase or word will be transformed into the text. The API will stop listening automatically after a couple of seconds of silence or when it is stopped manually.

Listing 6 - handling the event of pressing the stop button for converting speech to text

```
$('#pause-record-btn').on('click', function(e) {
  recognition.stop();
});
```

Convert text to speech. Speech synthesis is actually very simple. The API is accessible through the speechSynthesis object, and there are several methods for playing, pausing, and other things related to sound. There are also some interesting features that change pitch, speed, and even the reader's voice.

All that is needed for demonstration is the speak () method. It expects one argument - an instance of the SpeechSynthesisUtterance class.

Listing 7 shows all the code needed to read a line.

Listing 7 - reading text aloud with a computer synthesized voice

```
function readOutLoud(message) {
  var speech = new SpeechSynthesisUtterance();

  // Устанавливаем атрибуты текста и голоса.
  speech.text = message;
  speech.volume = 1;
  speech.rate = 1;
  speech.pitch = 1;

  window.speechSynthesis.speak(speech);
}
```

Conclusion. The article examined a method for converting human speech into text using the Web Speech API. A method for converting text into human speech by a computer-synthesized voice was also investigated.

In an era when voice assistants have become more popular than ever, a similar API provides a quick way to create bots that understand and speak human language.

Adding voice control to applications can also be a great way to improve accessibility. Visually impaired users can use both speech-to-text and text-to-speech user interfaces.

The speech synthesis and speech recognition APIs work quite well and easily handle various languages and accents. Unfortunately, at the moment they have limited browser support, which limits their use in production.

REFERENCES

1. Web Speech API – Web APIs | MDN [Electronic resource] - Access mode: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API - Access date: 24.02.2019.
2. HTML5 Speech Recognition API - codeburst [Electronic resource] - Access mode: https://codeburst.io/html5-speech-recognition-api-670846a50e92 - Access date: 24.02.2019.
3. Interfaces Web API | MDN [Electronic resource] - Access mode: https://developer.mozilla.org/ru/docs/Web/API - Access date: 25.02.2019.
4. Interface events [Electronic resource] – Access mode: https://learn.javascript.ru/event-details - Access date: 25.02.2019.