(function() {

if (!"speechSynthesis" in window) {

Document.write("Sorry. Your browser does not have speech support");

exit;

}

var synth = window.speechSynthesis;

console.log("speechSynthesis", synth);

// Test to see when the "onvoiceschanged" is triggered

synth.onvoiceschanged = function(e) {

console.log("onvoiceschanged triggered", e);

};

// Chrome loads voices asynchronously.

// Chrome and Edge loads the voices asynchronously (But not Firefox), so we have to  wait for them to appear

// Another way would be to use the "synth.onvoiceschanged" callback function. // But this have some drawbacks (In Chrome the event is triggered everytime you use  a Google voice), so Im using the timer method.

var waitTimerID = setInterval(function() {

console.log("ZZzz..");

// lang, default, name, voiceURI, localService

var voices = synth.getVoices();

// We got voices!

if (voices.length != 0) {

// Clear the interval timer, as we have the data needed

clearInterval(waitTimerID);

// Sort voices by language, then name (For Selectbox)

voices.sort(function(obj1, obj2) {

if (obj1.lang < obj2.lang) return -1;

if (obj1.lang > obj2.lang) return 1;

if (obj1.name < obj2.name) return -1;

if (obj1.name > obj2.name) return 1;

return 0;

});

var textInput = document.querySelector("#text");

var wordOutput = document.querySelector("#wordoutput");

var voicesSelect = document.querySelector("#voices");

var pitch = document.querySelector("#pitch");

var rate = document.querySelector("#rate");

var volume = document.querySelector("#volume");

var button = document.querySelector("#talk");

// Test if local storage is available

var lStorage = null;

if (

!"localStorage" in window ||

typeof window.localStorage !== "undefined"

) {

// Accessing "window.localStorage" on file: protocol, throws

a "SCRIPT16389: Unspecified error." in MS Edge.

// So it needs the extra "typeof" check.

lStorage = window.localStorage;

} else {

console.log("No localstorage support available!");

}

// Get word at specific position. Used for extracting the word currently

spoken

// Source: https://stackoverflow.com/questions/5173316/finding-the

word-at-a-position-in-javascript

var getWordAt = function(str, pos) {

// Perform type conversions.

str = String(str);

pos = Number(pos) >>> 0;

// Search for the word's beginning and end.

var left = str.slice(0, pos + 1).search(/\S+$/),

right = str.slice(pos).search(/\s/);

// The last word in the string is a special case.

// else Return the word, using the located bounds to extract it

from the string.

return right < 0 ? str.slice(left) : str.slice(left, right + pos);

};

// OnInput callback handler for range input controls

var onInput = function(e) {

e.target.previousElementSibling.previousElementSibling.innerHTML = e.target.value;

saveToLocalStorage();

};

// OnChange callback handler for voice selectbox

var onSelect = function(e) {

var curVoice = voices.find(

x => x.name ===

voicesSelect[voicesSelect.selectedIndex].value

);

// Reduce max rate if not a localService

var maxRate = curVoice.localService ? 10 : 2;

rate.setAttribute("max", maxRate);

var reset = 1;

pitch.value = reset;

pitch.previousElementSibling.previousElementSibling.innerHTML = reset; rate.value = reset;

rate.previousElementSibling.previousElementSibling.innerHTML = reset; volume.value = reset;

volume.previousElementSibling.previousElementSibling.innerHTML = reset; saveToLocalStorage();

setVoiceInfo();

};

// OnBoundary callback handler for when words is spoken

// Note: Event is not triggered when "localService" is false (Like the

Google voices in Chrome)

var onBoundary = function(e) {

if (e.name == "word") {

var word = getWordAt(e.target.text, e.charIndex);

wordOutput.innerHTML = word;

}

};

// OnEnd callback handler for when speaking ends

var onEnd = function(e) {

wordOutput.innerHTML = "";

};

// Trigger the SpeechSynthesisUtterance

var utterText = function() {

var curVoice = voices.find(

x => x.name ===

voicesSelect[voicesSelect.selectedIndex].value

);

// Cancel speaking if active

if (synth.speaking) synth.cancel();

var utterance = new SpeechSynthesisUtterance();

utterance.voice = curVoice;

utterance.lang = curVoice.lang;

utterance.pitch = pitch.value;

utterance.rate = rate.value;

utterance.volume = volume.value;

utterance.text = textInput.value;

if (curVoice.localService) utterance.onboundary =

onBoundary;

else {

wordOutput.innerHTML = " ";

}

utterance.onend = onEnd;

synth.speak(utterance);

};

var saveToLocalStorage = function() {

if (lStorage == null) return;

lStorage.setItem("voice",

voicesSelect[voicesSelect.selectedIndex].value);

lStorage.setItem("pitch", pitch.value);

lStorage.setItem("rate", rate.value);

lStorage.setItem("volume", volume.value);

};

/\* Add events to input controls (range sliders) and voice select \*/

volume.addEventListener("input", onInput);

pitch.addEventListener("input", onInput);

rate.addEventListener("input", onInput);

voicesSelect.addEventListener("change", onSelect);

button.addEventListener("click", utterText);

// Set initial position of range input sliders

var pVal = (rVal = vVal = 1);

if (lStorage !== null) {

pVal = lStorage.getItem("pitch") || 1;

rVal = lStorage.getItem("rate") || 1;

vVal = lStorage.getItem("volume") || 1;

}

pitch.value = pVal;

pitch.previousElementSibling.previousElementSibling.innerHTML =

pVal;

rate.value = rVal;

rate.previousElementSibling.previousElementSibling.innerHTML =

rVal;

volume.value = vVal;

volume.previousElementSibling.previousElementSibling.innerHTML = vVal;

// Remove existing items from selectbox

for (var i = voicesSelect.options.length - 1; i >= 0; i--) {

voicesSelect.remove(i);

}

// Populate selectbox

var voVal = lStorage !== null ? lStorage.getItem("voice") : "";

for (var i = 0; i < voices.length; i++) {

var option = document.createElement("option");

option.textContent = voices[i].name + " (" + voices[i].lang +

")";

option.value = voices[i].name;

voicesSelect.appendChild(option);

if (voVal == voices[i].name) voicesSelect.selectedIndex = i;

}

setVoiceInfo();

// Reduce max rate if voice is not a localService

var curVoice = voices.find(

x => x.name ===

voicesSelect[voicesSelect.selectedIndex].value

);

var maxRate = curVoice.localService ? 10 : 2;

rate.setAttribute("max", maxRate);

}

}, 10);

})();