accusoft®

Adding Text-to-Speech to Your Document Viewer

Featured

Jared Jacoby
Software Engineer, PrizmDoc Viewer

Text-to-speech (also known as speech synthesis) is the artificial production of human speech. It is most commonly used to allow people with visual impairments or reading disabilities to listen to written words. Thanks to the Speech Synthesis API, which is natively supported in modern browsers, text-to-speech functionality can easily be integrated into a web application. In this post, I will outline how to add it to the PrizmDoc Viewer with a few simple steps.

accusoft®

Achieving True Security in a Cloud-Hosted World

1. Add a Listen Menu

Add the code below to viewer-assets/templates/viewerTemplate.html to add a “Listen” menu to the viewer containing a play, pause, and stop button. You can add this code after the div containing “data-pcc-nav-tab=”esign”” to show the “Listen” tab to the right of the existing “E-Sign” tab.

```
<div class="pcc-tab" data-pcc-nav-tab="listen" data-pcc-removable-id="listenTab">
  <div class="pcc-tab-item">
    <span class="pcc-icon pcc-icon-listen"></span><%= listen %>
  </div>
  <div class="pcc-tab-pane">
    <div class="pcc-left">
      <button class="pcc-icon pcc-disabled" title="<%= stop %>" data-pcc-listen="stop">
        <span>&#9724;</span>
      </button>
      <button class="pcc-icon pcc-disabled" title="<%= pause %>" data-pcc-listen="pause">
        <span>&#10074;&#10074;</span>
      </button>
      <button class="pcc-icon pcc-disabled" title="<%= play %>" data-pcc-listen="play">
        <span>&#9658;</span>
      </button>
    </div>
  </div>
</div>
```

To show an icon on the new “Listen” tab, create a new file (named listen.svg) in the viewer-assets/icons/svg folder, and add the SVG below to the file. This icon is referenced as pcc-icon-listen in the HTML template code above.

```
<svg width="52" height="52" >
  <path d="m21,20l10,7l-10,7l0,-14z"/>
  <path d="m38.681,17.159c-0.694,-0.947 -1.662,-2.053 -2.724,-3.116s-2.169,-2.03
-3.116,-2.724c-1.612,-1.182 -2.393,-1.319 -2.841,-1.319l-15.5,0c-1.378,0 -2.5,1.121
-2.5,2.5l0,27c0,1.378 1.122,2.5 2.5,2.5l23,0c1.378,0 2.5,-1.122 2.5,-2.5l0,-19.5c0,-0.448
-0.137,-1.23 -1.319,-2.841zm-4.138,-1.702c0.959,0.959 1.712,1.825 2.268,2.543l-4.811,0l0,-
4.811c0.718,0.556 1.584,1.309 2.543,2.268zm3.457,24.043c0,0.271 -0.229,0.5
-0.5,0.5l-23,0c-0.271,0 -0.5,-0.229 -0.5,-0.5l0,-27c0,-0.271 0.229,-0.5 0.5,-0.5c0,0 15.499,0
15.5,0l0,7c0,0.552 0.448,1 1,1l7,0l0,19.5z"/>
</svg>
```

The HTML template code above also references some text. All viewer text is stored in the language file, viewer-assets/languages/en-US.json. You will need to add the following new terms to the language file.

```
"listen": "Listen",
"stop": "Stop",
"pause": "Pause",
"play": "Play",
```

You will need to build the viewer after making the above changes, and copy the output dist/viewer-assets files over the viewer-assets files of your web page. Steps for building the viewer are in the README file of each installed PrizmDoc Viewer sample. In summary, you will need to run “npm install” and then run “gulp build”. You will need to copy all of the files in the viewer-assets folder except Gulpfile.js and package.json to a “src” folder in the viewer-assets folder first.

2. Reference the UI and Enable in Supported Browsers

The code in this step and the following steps can be added to the function where you are calling the pccViewer method to create the viewer. For example, in the webforms sample you would add it to the embedViewer function in Default.aspx.

Add variables to reference the stop, pause, and play buttons. Enable the play button only if the browser supports Speech Synthesis. The Speech Synthesis API is natively supported by the latest versions of Chrome, Firefox, Edge, and Safari, but is not supported in Internet Explorer 11 and many mobile browsers.

```
function embedViewer(options) {
  var viewer = $('#viewer1').pccViewer(options);

  var $speechStop = $('#viewer1').find("[data-pcc-listen=stop]");
  var $speechPause = $('#viewer1').find("[data-pcc-listen=pause]");
  var $speechPlay = $('#viewer1').find("[data-pcc-listen=play]");

  if ('speechSynthesis' in window) {
    $speechPlay.prop('disabled', false).removeClass('pcc-disabled');
  }
```

3. Implement Speaking

Add the play button click event handler below. Clicking the play button will begin playing all of the pages starting at the current page, or resume if playing had already started and was paused.

```
var startedPlaying = false;
$speechPlay.on('click', function (ev) {
  $speechStop.prop('disabled', false).removeClass('pcc-disabled');
  $speechPause.prop('disabled', false).removeClass('pcc-disabled');
  $speechPlay.prop('disabled', true).addClass('pcc-disabled');
  if (startedPlaying === false) {
    startedPlaying = true;
    play(viewer.viewerControl.getPageNumber());
  }
  else {
    window.speechSynthesis.resume();
  }
});
```

Add the play and stop functions below. The play function requests the text of a page and uses it to create a SpeechSynthesisUtterance. The speechSynthesis.speak method is called to initiate the speaking. When completed, the onend event handler is used to play the next page, unless the last page has been played. In that case, the stop function is called to reset the state to allow playing again.

```
function play(pageNumber) {
  viewer.viewerControl.setPageNumber(pageNumber);
  viewer.viewerControl.requestPageText(pageNumber).then(
    function (pageText) {
      // Lines in documents may end in carriage returns. Remove them from
      // the text to avoid pauses in the speech.
      pageText = pageText.replace(/(rn|n|r|_)/gm, ' ');
      var utterance = new window.SpeechSynthesisUtterance(pageText);
      // To work around a known bug in Chrome with onend not always firing,
      // add a global reference to the utterance
      // https://bugs.chromium.org/p/chromium/issues/detail?id=509488
      window.utterances = [];
      window.utterances.push(utterance);
      utterance.onend = function () {
        if (startedPlaying) {
          if (pageNumber >= viewer.viewerControl.getPageCount()) {
            stop();
          }
          else {
            play(pageNumber + 1);
          }
        }
      };
      window.speechSynthesis.cancel();
      window.speechSynthesis.speak(utterance);
    }
  );
}

function stop() {
  startedPlaying = false;
  window.speechSynthesis.cancel();
  $speechStop.prop('disabled', true).addClass('pcc-disabled');
  $speechPause.prop('disabled', true).addClass('pcc-disabled');
  $speechPlay.prop('disabled', false).removeClass('pcc-disabled');
}
```

4. Implement Pausing

The speechSynthesis.pause method pauses the speech. Add the following pause button click event handler after the previously added play button click event handler.

```
$speechPause.on('click', function (ev) {
  $speechPause.prop('disabled', true).addClass('pcc-disabled');
  $speechPlay.prop('disabled', false).removeClass('pcc-disabled');
  window.speechSynthesis.pause();
});
```

5. Implement Stopping

To implement stopping of the speech, add the following stop button click event handler after the previously added Pause button click event handler.

```
$speechStop.on('click', function (ev) {
  stop();
});
```

Once you've followed these steps to integrate text-to-speech functionality into your PrizmDoc Viewer, your users can start listening to documents with the click of a button. For more information on what you can do with PrizmDoc Viewer, visit the documentation.