Player

Basic setup

Now that we have the basic component ready, it is time to add some sound. AlphaTab comes with an audio synthesis engine named AlphaSynth to play the songs via MIDI and a SoundFont2/SoundFont3 file.

To get the player configured is also quite easy:

1. We tell alphaTab that we want the player active and where to find the SoundFont file that should be used. alphaTab ships a free SoundFont file for usage on your website.
2. We tell alphaTab which DOM element to scroll when the cursor moves.
3. We tell alphaTab how the cursors should look like
4. We add some UI elements for the user to start/pause the playback and show the current position.

Initialize Player | **Cursor Styles**

```
const settings = {
  file: "https://www.alphatab.net/files/canon.gp",
  player: {
      enablePlayer: true,
      soundFont:
'https://cdn.jsdelivr.net/npm/@coderline/alphatab@latest/dist/soundfont/soni
      scrollElement: wrapper.querySelector('.at-viewport') // this is the el
scroll during playback
  }
};
const api = new alphaTab.AlphaTabApi(main, settings);
```

Loading indicator

Just like alphaTab, alphaSynth might need some time to initialize and load all required data. Especially the SoundFonts can become quite big and might need some time to load. So we will add a small loading indicator to our footer so that the user will know once the player is ready for usage.

It is better to have a separate loading indicator for the player. A typical user will look at the music sheet for a bit before pressing play. If we keep the main loading indicator for the music sheet visible until the player is ready, the user does not get the chance to have a look.

To keep the tutorial simple we will just show a percentage value but you might want to show a nice looking progress indicator on your site.

New DOM **New JavaScript**

```
<div class="at-controls">
  <div class="at-controls-left">
    <span class="at-player-progress">0%</span>
...
```

First we add a placeholder in which we will show the percentage value. Then we will use the `soundFontLoad` event to indicate the load progress of the (usually larger) SoundFont. The `playerReady` will tell us when to hide the indicator.

Main player controls

Now we need to add the required controls for the users to play and pause the audio. We add one button for play/pause which will change the icon depending on the current playback state. We also add one button which stops the playback and moves the cursor back to the start.

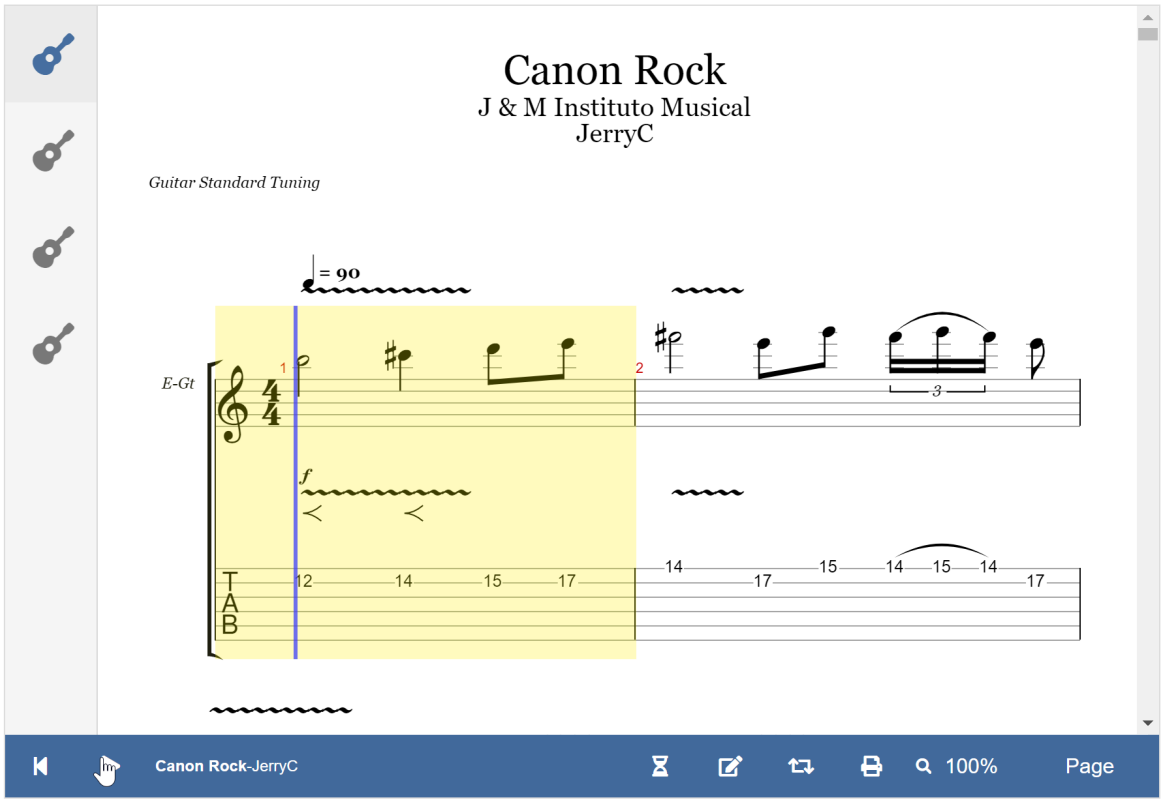
We use `playPause()` and `stop()` to send the right signals to alphaTab and we use the `playerStateChanged` event to update the button icon. To ensure that the user does not interact with the player before it is ready, we keep the buttons disabled until the player is ready.

New DOM New CSS New JavaScript

Here we have the two new buttons:

```
<div class="at-controls">
  <div class="at-controls-left">
    <a class="btn at-player-stop disabled">
      <i class="fas fa-step-backward"></i>
    </a>
    <a class="btn at-player-play-pause disabled">
      <i class="fas fa-play"></i>
    </a>
    <span class="at-player-progress">0%</span>
...
```

The result looks like this:



Showing the current time

In this step we will add a textual indicator of the current position of the playback. You could use the provided information to build a progress bar indicator for the users which could even be interactive with seeking, but this tutorial will not cover that.

As in most other steps we will add some new DOM elements, corresponding CSS and then hook it up with the alphaTab event, `playerPositionChanged`, providing the necessary data.

New DOM **New JavaScript**

```
...
<div class="at-song-info">
  <span class="at-song-title"></span> -
  <span class="at-song-artist"></span>
</div>
<div class="at-song-position">00:00 / 00:00</div>
...
```

Final File

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>AlphaTab Tutorial</title>
    <script
src="https://cdn.jsdelivr.net/npm/@coderline/alphatab@latest/dist/alphaTab.j
</script>
    <script src="https://kit.fontawesome.com/b43f0e512e.js"></script>
    <style type="text/css">
      body {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 12px;
      }

      .at-wrap {
        width: 80vw;
        height: 80vh;
        margin: 0 auto;
```

```
  border: 1px solid rgba(0, 0, 0, 0.12);
  display: flex;
  flex-direction: column;
  overflow: hidden;
  position: relative;
}

.at-content {
  position: relative;
  overflow: hidden;
  flex: 1 1 auto;
}

/** Sidebar **/
.at-sidebar {
  position: absolute;
  top: 0;
  left: 0;
  bottom: 0;
  max-width: 70px;
  width: auto;
  display: flex;
  align-content: stretch;
  z-index: 1001;
  overflow: hidden;
  border-right: 1px solid rgba(0, 0, 0, 0.12);
  background: #f7f7f7;
}

.at-sidebar:hover {
  max-width: 400px;
  transition: max-width 0.2s;
  overflow-y: auto;
}

.at-viewport {
  overflow-y: auto;
  position: absolute;
  top: 0;
  left: 70px;
  right: 0;
  bottom: 0;
  padding-right: 20px;
}
```

```css
.at-footer {
  flex: 0 0 auto;
  background: #436d9d;
  color: #fff;
}

/** Overlay **/

.at-overlay {
  /** Fill Parent */
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  z-index: 1002;

  /* Blurry dark shade */
  backdrop-filter: blur(3px);
  background: rgba(0, 0, 0, 0.5);

  /** center content */
  display: flex;
  justify-content: center;
  align-items: flex-start;
}

.at-overlay-content {
  /* white box with drop-shadow */
  margin-top: 20px;
  background: #fff;
  box-shadow: 0px 5px 10px 0px rgba(0, 0, 0, 0.3);
  padding: 10px;
}

/** Track selector **/
.at-track {
  display: flex;
  position: relative;
  padding: 5px;
  transition: background 0.2s;
  cursor: pointer;
}
```

```css
.at-track:hover {
  background: rgba(0, 0, 0, 0.1);
}

.at-track > .at-track-icon,
.at-track > .at-track-details {
  display: flex;
  flex-direction: column;
  justify-content: center;
}

.at-track > .at-track-icon {
  flex-shrink: 0;
  font-size: 32px;
  opacity: 0.5;
  transition: opacity 0.2s;
  width: 64px;
  height: 64px;
  margin-right: 5px;
  align-items: center;
}

.at-track-name {
  font-weight: bold;
  margin-bottom: 5px;
}

.at-track:hover > .at-track-icon {
  opacity: 0.8;
}

.at-track.active {
  background: rgba(0, 0, 0, 0.03);
}

.at-track.active > .at-track-icon {
  color: #4972a1;
  opacity: 1;
}

.at-track > .at-track-name {
  font-weight: 500;
}
```

```
/** Footer **/
.at-controls {
  flex: 0 0 auto;
  display: flex;
  justify-content: space-between;
  background: #436d9d;
  color: #fff;
}

.at-controls > div {
  display: flex;
  justify-content: flex-start;
  align-content: center;
  align-items: center;
}

.at-controls > div > * {
  display: flex;
  text-align: center;
  align-items: center;
  justify-content: center;
  cursor: pointer;
  padding: 4px;
  margin: 0 3px;
}

.at-controls .btn {
  color: #fff;
  border-radius: 0;
  height: 40px;
  width: 40px;
  height: 40px;
  font-size: 16px;
}
.at-controls .btn.disabled {
  cursor: progress;
  opacity: 0.5;
}

.at-controls a.active {
  background: #5588c7;
  text-decoration: none;
}
```

```css
.at-controls .btn i {
  vertical-align: top;
}

.at-controls select {
  -moz-appearance: none;
  -webkit-appearance: none;
  appearance: none;
  border: none;
  width: 100%;
  height: 40px;
  background: #436d9d;
  padding: 4px 10px;
  color: #fff;
  font-size: 16px;
  text-align-last: center;
  text-align: center;
  -ms-text-align-last: center;
  -moz-text-align-last: center;
  cursor: pointer;
}

.at-song-title {
  font-weight: bold;
}

.at-cursor-bar {
  /* Defines the color of the bar background when a bar is played */
  background: rgba(255, 242, 0, 0.25);
}

.at-selection div {
  /* Defines the color of the selection background */
  background: rgba(64, 64, 255, 0.1);
}

.at-cursor-beat {
  /* Defines the beat cursor */
  background: rgba(64, 64, 255, 0.75);
  width: 3px;
}

.at-highlight * {
```

```
        /* Defines the color of the music symbols when they are being played
        fill: #0078ff;
        stroke: #0078ff;
      }
    </style>
  </head>

  <body>
    <div class="at-wrap">
      <div class="at-overlay">
        <div class="at-overlay-content">
          Music sheet is loading
        </div>
      </div>
      <div class="at-content">
        <div class="at-sidebar">
          <div class="at-sidebar-content">
            <div class="at-track-list"></div>
          </div>
        </div>
        <div class="at-viewport">
          <div class="at-main"></div>
        </div>
      </div>
      <div class="at-controls">
        <div class="at-controls-left">
          <a class="btn at-player-stop disabled">
            <i class="fas fa-step-backward"></i>
          </a>
          <a class="btn at-player-play-pause disabled">
            <i class="fas fa-play"></i>
          </a>
          <span class="at-player-progress">0%</span>
          <div class="at-song-info">
            <span class="at-song-title"></span> -
            <span class="at-song-artist"></span>
          </div>
          <div class="at-song-position">00:00 / 00:00</div>
        </div>
        <div class="at-controls-right">
          <a class="btn toggle at-count-in">
            <i class="fas fa-hourglass-half"></i>
          </a>
          <a class="btn at-metronome">
```

```
            <i class="fas fa-edit"></i>
          </a>
          <a class="btn at-loop">
            <i class="fas fa-retweet"></i>
          </a>
          <a class="btn at-print">
            <i class="fas fa-print"></i>
          </a>
          <div class="at-zoom">
            <i class="fas fa-search"></i>
            <select>
              <option value="25">25%</option>
              <option value="50">50%</option>
              <option value="75">75%</option>
              <option value="90">90%</option>
              <option value="100" selected>100%</option>
              <option value="110">110%</option>
              <option value="125">125%</option>
              <option value="150">150%</option>
              <option value="200">200%</option>
            </select>
          </div>
          <div class="at-layout">
            <select>
              <option value="horizontal">Horizontal</option>
              <option value="page" selected>Page</option>
            </select>
          </div>
        </div>
      </div>
    </div>

    <template id="at-track-template">
      <div class="at-track">
        <div class="at-track-icon">
          <i class="fas fa-guitar"></i>
        </div>
        <div class="at-track-details">
          <div class="at-track-name"></div>
        </div>
      </div>
    </template>

    <script type="text/javascript">
```

```
        // load elements
        const wrapper = document.querySelector(".at-wrap");
        const main = wrapper.querySelector(".at-main");

        // initialize alphatab
        const settings = {
          file: "https://www.alphatab.net/files/canon.gp",
          player: {
            enablePlayer: true,
            soundFont:
"https://cdn.jsdelivr.net/npm/@coderline/alphatab@latest/dist/soundfont/soni
            scrollElement: wrapper.querySelector('.at-viewport')
          },
        };
        const api = new alphaTab.AlphaTabApi(main, settings);

        // overlay logic
        const overlay = wrapper.querySelector(".at-overlay");
        api.renderStarted.on(() => {
          overlay.style.display = "flex";
        });
        api.renderFinished.on(() => {
          overlay.style.display = "none";
        });

        // track selector
        function createTrackItem(track) {
          const trackItem = document
            .querySelector("#at-track-template")
            .content.cloneNode(true).firstElementChild;
          trackItem.querySelector(".at-track-name").innerText = track.name;
          trackItem.track = track;
          trackItem.onclick = (e) => {
            e.stopPropagation();
            api.renderTracks([track]);
          };
          return trackItem;
        }
        const trackList = wrapper.querySelector(".at-track-list");
        api.scoreLoaded.on((score) => {
          // clear items
          trackList.innerHTML = "";
          // generate a track item for all tracks of the score
          score.tracks.forEach((track) => {
```

```
    trackList.appendChild(createTrackItem(track));
  });
});
api.renderStarted.on(() => {
  // collect tracks being rendered
  const tracks = new Map();
  api.tracks.forEach((t) => {
    tracks.set(t.index, t);
  });
  // mark the item as active or not
  const trackItems = trackList.querySelectorAll(".at-track");
  trackItems.forEach((trackItem) => {
    if (tracks.has(trackItem.track.index)) {
      trackItem.classList.add("active");
    } else {
      trackItem.classList.remove("active");
    }
  });
});

/** Controls **/
api.scoreLoaded.on((score) => {
  wrapper.querySelector(".at-song-title").innerText = score.title;
  wrapper.querySelector(".at-song-artist").innerText = score.artist;
});

const countIn = wrapper.querySelector('.at-controls .at-count-in');
countIn.onclick = () => {
  countIn.classList.toggle('active');
  if (countIn.classList.contains('active')) {
    api.countInVolume = 1;
  } else {
    api.countInVolume = 0;
  }
};

const metronome = wrapper.querySelector(".at-controls .at-metronome");
metronome.onclick = () => {
  metronome.classList.toggle("active");
  if (metronome.classList.contains("active")) {
    api.metronomeVolume = 1;
  } else {
    api.metronomeVolume = 0;
  }
```

```
};

const loop = wrapper.querySelector(".at-controls .at-loop");
loop.onclick = () => {
  loop.classList.toggle("active");
  api.isLooping = loop.classList.contains("active");
};

wrapper.querySelector(".at-controls .at-print").onclick = () => {
  api.print();
};

const zoom = wrapper.querySelector(".at-controls .at-zoom select");
zoom.onchange = () => {
  const zoomLevel = parseInt(zoom.value) / 100;
  api.settings.display.scale = zoomLevel;
  api.updateSettings();
  api.render();
};

const layout = wrapper.querySelector(".at-controls .at-layout select")
layout.onchange = () => {
  switch (layout.value) {
    case "horizontal":
      api.settings.display.layoutMode = alphaTab.LayoutMode.Horizontal
      break;
    case "page":
      api.settings.display.layoutMode = alphaTab.LayoutMode.Page;
      break;
  }
  api.updateSettings();
  api.render();
};

// player loading indicator
const playerIndicator = wrapper.querySelector(
  ".at-controls .at-player-progress"
);
api.soundFontLoad.on((e) => {
  const percentage = Math.floor((e.loaded / e.total) * 100);
  playerIndicator.innerText = percentage + "%";
});
api.playerReady.on(() => {
  playerIndicator.style.display = "none";
```

```
});

// main player controls
const playPause = wrapper.querySelector(
  ".at-controls .at-player-play-pause"
);
const stop = wrapper.querySelector(".at-controls .at-player-stop");
playPause.onclick = (e) => {
  if (e.target.classList.contains("disabled")) {
    return;
  }
  api.playPause();
};
stop.onclick = (e) => {
  if (e.target.classList.contains("disabled")) {
    return;
  }
  api.stop();
};
api.playerReady.on(() => {
  playPause.classList.remove("disabled");
  stop.classList.remove("disabled");
});
api.playerStateChanged.on((e) => {
  const icon = playPause.querySelector("i.fas");
  if (e.state === alphaTab.synth.PlayerState.Playing) {
    icon.classList.remove("fa-play");
    icon.classList.add("fa-pause");
  } else {
    icon.classList.remove("fa-pause");
    icon.classList.add("fa-play");
  }
});

// song position
function formatDuration(milliseconds) {
  let seconds = milliseconds / 1000;
  const minutes = (seconds / 60) | 0;
  seconds = (seconds - minutes * 60) | 0;
  return (
    String(minutes).padStart(2, "0") +
    ":" +
    String(seconds).padStart(2, "0")
  );
```

```
        }

        const songPosition = wrapper.querySelector(".at-song-position");
        let previousTime = -1;
        api.playerPositionChanged.on((e) => {
          // reduce number of UI updates to second changes.
          const currentSeconds = (e.currentTime / 1000) | 0;
          if (currentSeconds == previousTime) {
            return;
          }

          songPosition.innerText =
            formatDuration(e.currentTime) + " / " + formatDuration(e.endTime);
        });
      </script>
    </body>
  </html>
```