More Controls

In the following steps of the tutorial we will add some basic controls for the end users which will allow them to:

- See the song title and artist
- Enable/Disable the count-in
- Enable/Disable the metronome
- Enable/Disable looping
- Print the music sheet
- Change the zoom level
- Change the layout of the music sheet

Extend footer

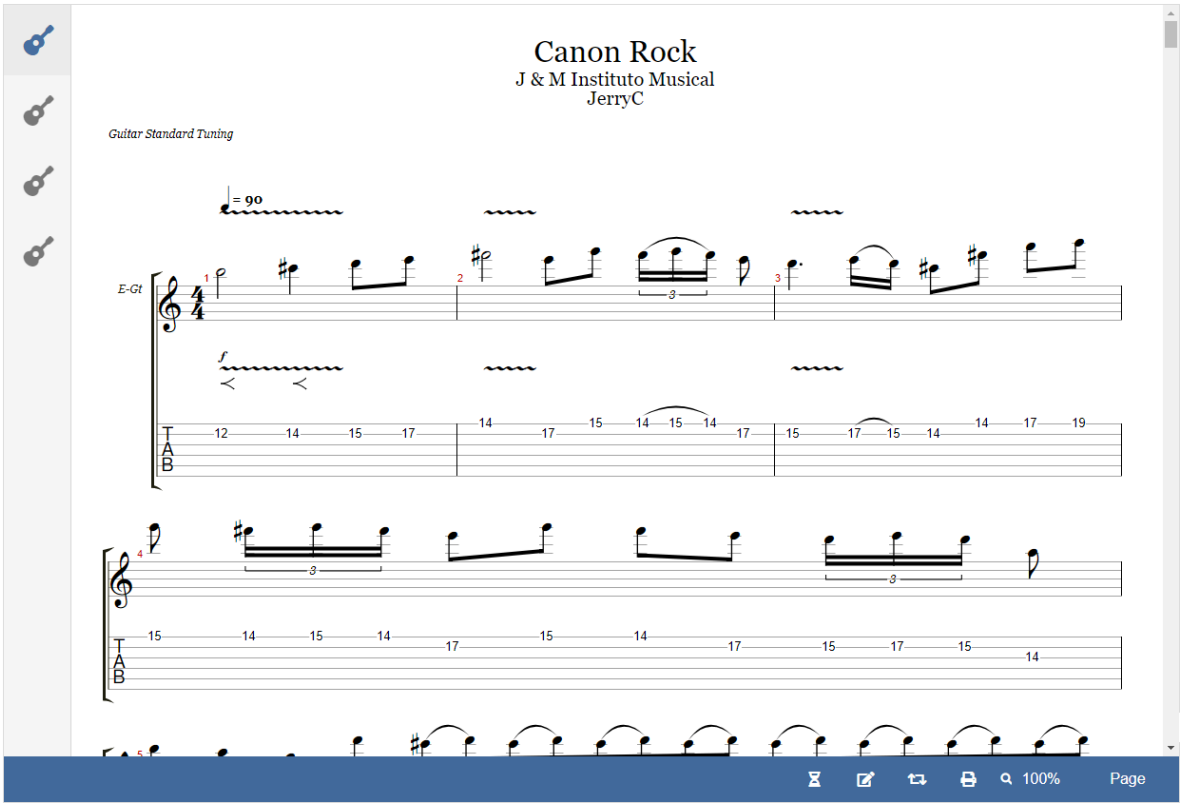
As a start we will build the DOM of our footer which will act as toolbar. This is once again just some regular web development, so we will go through this rather quickly:

New DOM New CSS

```
<div class="at-controls">
  <div class="at-controls-left">
    <div class="at-song-info">
      <span class="at-song-title"></span> -
      <span class="at-song-artist"></span>
    </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 toggle at-metronome">
      <i class="fas fa-edit"></i>
```

```
    </a>
    <a class="btn toggle 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>
```

The code above creates some placeholder elements, buttons and selections for the user to interact with. The next step is to add some listeners to the user interaction and tell alphaTab what to do.



Song Info

To fill the song info we will again subscribe to the `scoreLoaded` event which we used to fill the track selector. Then we just select the placeholder elements and fill them with the appropriate values from the data model:

```
api.scoreLoaded.on((score)=>{
  wrapper.querySelector('.at-song-title').innerText = score.title;
  wrapper.querySelector('.at-song-artist').innerText = score.artist;
});
```

Count-In and Metronome

The count-in and metronome will become relevant when we activate the player in the next step, but we can already create the controls which will set their volumes based on their toggle button states:

```
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;
  }
};
```

We toggle the `active` CSS class as a visual indication whether the feature is active or not. We then check the visual state to tell alphaTab whether the feature should be active or not by setting the `countInVolume` and `metronomeVolume` values respectively.

Looping

Similar to the count-in and metronome, we use the `active` class to decide whether the looping should be active using the `isLooping` option. This will also only be relevant once we have the player feature enabled:

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

Printing

Sometimes users might want to print out the music sheet or save it as a PDF. Due to the dynamic rendering that alphaTab performs a simple CTRL+P in the browser might not bring the desired results. The whole music sheet is usually rendered in the background, and there are special features that only show the visible portions in the viewport.

When using the default browser printing you might just see an empty space. The alphaTab API provides a `print` function which optimizes the rendering for printing and opens it in a popup window.

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

Zoom

As alphaTab can change the scale in which the music sheet is rendered, we offer the user a select box to change this scale.

```
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();
};
```

Updating the zoom level is a bit more effort than the previous tasks but still not very complicated:
First we fill the settings with the new scale. Then we tell alphaTab to load in these updated settings and re-render the music sheet.

Layout

alphaTab can either render the music sheet in a page-like fashion that grows from top to bottom along the available width, or it can show the music sheet in a horizontal scrolling

fashion.

These options are also offered to the user via a select box. To apply the user selection we forward the user input to the settings object and trigger an update, just like we did for the zoom level:

```
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();
};
```

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;
}

.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;
}

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

.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;
    }
  </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">
        <div class="at-song-info">
          <span class="at-song-title"></span> -
          <span class="at-song-artist"></span>
        </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",
  };
  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();
    };
  </script>
</body>

</html>
```