Exercise 3: HTML5, CSS and JavaScript

HTML5, CSS3 and JavaScript form together a dynamic trio. You can build astonishing things with little coding if can utilize them efficiently.

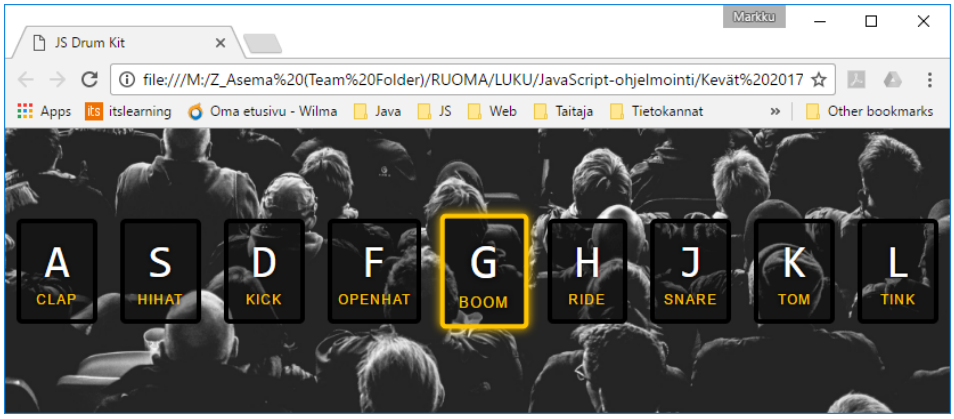
In this exercise you'll learn how to implement some interesting functionalities with just HTML5 and CSS with little programming.

DOM implementation varies between different browsers which needs to be taken into account when writing browser JavaScript programs. Additionally there are several versions of the same browser in active use at the same time and the supported features differ between browser versions too.

One also needs to check the browser compatibility especially for some more recent HTML5- ja CSS-features too. You can check them for example at the site http://caniuse.com . On the site http://doiuse.herokuapp.com/ you can check if you are using features that may not work properly on commonly used browsers.

Programmins exercise

Your task is to implement a drumset that can be played using the keyboard:



You have two functionalities to implement:

1. playing a designated sound when a key is pressed
2. visualization for a key press (see above)

Detecting key presses

First you need to catch the key presses. Borwser key presses generate keyboard events. Add a event listener to the document object to catch the appropriate events (keydown). Your event handler can at first just output the event information to the console. From the event properties you can find that the key pressed can be identified by the value of attribute keyCode.

```
document.addEventListener('keydown', playSound);
```

Above `playSound` is the event handler for keyDown. Note that keyboard events belong the the whole document, not any single element. The function definition could be e.g.:

```
function playSound(event) {
  console.log(event.keyCode);
}
```

Playing a sound

The next task is to make the function play sounds. HTML5 tag `audio` is already capable of playing sound files and we have readymake sound files for each instrument. The elements in the template already have the HTML attribute `data-key` with the correct key code as value.

From now on you can proceed the same way as before; look up the correct audio DOM object and call the method play(). No additional player components or audio libraries are needed, your browser can do it on its own.

You can find the audio element that corresponds to the key pressed like this:

```
var selector = 'audio[data-key="' + keyCode + '"] ';
var audio = document.querySelector(selector);
```

You construct a selector to look up the audio element that has attribute data-key with value keyCode. KeyCode is your variable, assign the key code of the pressed key to it.

However, there are no audio elements for all the possible key codes. If some other key than one of your drum kit keys is pressed the audio variable will be null. The the variable is not null we know that one of the drum kit keys was pressed an dyou should rewind the sound file (why?) and play it:

```
if (audio !== null) {
  audio.currentTime = 0;
  audio.play();
}
```

If everything worked out as planned your drum kit should be playable.

Visual effect

The key presses can be visualized shrewdly by just modifying previously defined CSS styles. All key elements have class key. The effect can be implemented just adding another class playing to the active element and removeing it later. You can try it in the browser.

The dynamic effect can be implemented with rule:

```
transition: all .07s ease;
```

Lengthen the transition time to be able to see what actually happens. When the transition is complete the element generates event `transitionend`. When that occurs you can remove the class playing and restore the key to it's original status.

Implement this: then a key is pressed, get the correct DOM element for the key and add class playing to it.:

```
key.classList.add('playing');
```

Add an event listener to all key elements to handle the transitionend event:

```
var keys = document.querySelectorAll('.key');
keys.forEach(function(key) {
    key.addEventListener('transitionend', revertStyle);
```

```
}
```

You also need to define function `revertStyle`:

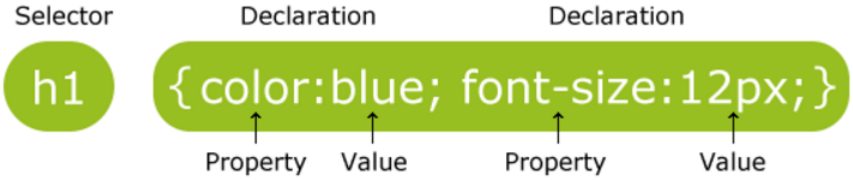
```
function revertStyle(event) {
  if (event.propertyName === 'transform') {
    this.classList.remove('playing');
  }
}
```

If everything worked out the visualizations should work too.

Did that feel too lengthy and complicated?

This exercise is written by Wes Bos and it is part of a free JavaScript course called JavaScript30. You can also look up the instruction video on YouTube and do this exercise with Wes Bos himself.

Questions



Picture 1 CSS rule syntax (http://www.w3schools.com/css/css_syntax.asp)

1. Explain which elements the following CSS selectors select:

   ```
   body,html {…}
   .keys {…}
   audio[data-key="65"] {…}
   .key[data-key=”71” {…}
   ```

   Explain also the selectors from the first exercise:

   ```
   #laskuri {…}
   * {…}
   ```

2. In this exercise we used methods `document.querySelector()` and `document.querySelectorAll()` to look up DOM elements from document instead of the familiar `document.getElementById()` ja `getElementsByClass`. How do they work?
3. Test your drum kit with at least three different browsers and report your findings. Hint: A test report should include at least which borwser and browser versions was used in testing and if there were problems what were they and how can you make them happen.
4. HTML tag `audio` also can provide music player controls (play, stip, pause etc.). Try adding a music player to a HTML page and provide a link to the page.
5. List at least three interesting HTML5 tags and describe briefly what you can do with them.

Bonus

Challenge yourself to get to the next level!

If you tried your drum kit on a mobile phone you probably noticed that it does not work there since the phone does not have a keyboard and there are no keyboard events. Make it usable on mobile phone browsers too e.g. by adding support for mouse events too.

Submission

- Document with the name(s) of the author(s) and answers to all the questions
- Link a web page that contains the working program

Assess your learning and give feedback about the exercise in the text field on the submission form.

Hints

- If you read the text you already notices that you can look up an instuction video on YouTube. The author is Wes Bos and the exercise belongs to a free JavaScript course called JavaScipt30.
- You can find information about CSS selectors for example in W3Schools http://www.w3schools.com/css/default.asp
- Another good reference is http://www.quackit.com/
- Here's an article about the audio tag: https://www.htmlgoodies.com/primers/html/article.php/3920991/HTML5-Primer-How-To-Use-the-Audio-Tag.htm