JavaScript Drum Kit Explanation

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JS Drum Kit</title>
  <link rel="stylesheet" href="style.css">
  <link rel="icon" href="https://fav.farm/✍" />
  <style>
    body {
      background-repeat: no-repeat;
      background-size: cover;
    }
  </style>
</head>
<body>
  <div class="keys">
    <div data-key="65" class="key">
      <kbd>A</kbd>
      <span class="sound">clap</span>
    </div>
    <div data-key="83" class="key">
      <kbd>S</kbd>
      <span class="sound">hihat</span>
    </div>
    <div data-key="68" class="key">
      <kbd>D</kbd>
      <span class="sound">kick</span>
    </div>
    <div data-key="70" class="key">
      <kbd>F</kbd>
      <span class="sound">openhat</span>
    </div>
    <div data-key="71" class="key">
      <kbd>G</kbd>
      <span class="sound">boom</span>
    </div>
    <div data-key="72" class="key">
      <kbd>H</kbd>
      <span class="sound">ride</span>
    </div>
    <div data-key="74" class="key">
      <kbd>J</kbd>
      <span class="sound">snare</span>
    </div>
    <div data-key="75" class="key">
```

```
      <kbd>K</kbd>
      <span class="sound">tom</span>
    </div>
    <div data-key="76" class="key">
      <kbd>L</kbd>
      <span class="sound">tink</span>
    </div>
  </div>

  <audio data-key="65" src="sounds/clap.wav"></audio>
  <audio data-key="83" src="sounds/hihat.wav"></audio>
  <audio data-key="68" src="sounds/kick.wav"></audio>
  <audio data-key="70" src="sounds/openhat.wav"></audio>
  <audio data-key="71" src="sounds/boom.wav"></audio>
  <audio data-key="72" src="sounds/ride.wav"></audio>
  <audio data-key="74" src="sounds/snare.wav"></audio>
  <audio data-key="75" src="sounds/tom.wav"></audio>
  <audio data-key="76" src="sounds/tink.wav"></audio>

<script>
  function removeTransition(e) {
  if (e.propertyName !== 'transform') return;
  e.target.classList.remove('playing');
}

let keyDown = false; // To keep track of key press state

function playSound(e) {
  if (keyDown) return; // Exit if key is already pressed down
  const audio = document.querySelector(`audio[data-key="$
{e.keyCode}"]`);
  const key = document.querySelector(`div[data-key="${e.keyCode}"]`);
  if (!audio) return;

  keyDown = true; // Set key press state to true
  key.classList.add('playing');
  audio.currentTime = 0;
  audio.play();

  setTimeout(() => {
    keyDown = false; // Set key press state back to false
  }, 200); // 200 milliseconds (0.2 seconds)
}

const keys = Array.from(document.querySelectorAll('.key'));
keys.forEach(key => {
  key.addEventListener('transitionend', removeTransition);
  key.addEventListener('keyup', () => {
    keyDown = false; // Set key press state to false on keyup event
```

```
  });
});

window.addEventListener('keydown', playSound);
</script>
```

Below is a brief explanation of how the drum kit is build:

1. Firstly, there is a div container with the class "keys" that contains the keys of the drum kit. Each key is represented by a div element with the class "key". Within each "key," the letter of the key (e.g., A, S, D) is displayed using the kbd tag, and the sound name (e.g., clap, hihat) is displayed using the span tag.

2. Below the div container with the class "keys," there are audio components for each sound. Each audio element is associated with a specific key using the data-key attribute, which contains the corresponding key code. For example, the first audio tag has data-key="65," which corresponds to the letter 'A' (keyCode 65) of the key.

3. After the HTML markup, the JavaScript code follows. The script tag encapsulates the JavaScript functionality. The code makes use of several functions defined in JavaScript.

4. The function "removeTransition" is defined. This function is used to remove the visual transition effects from a key after the transition is completed. It checks if the property type (propertyName) of the transition is 'transform' and then removes the 'playing' class from the respective element.

5. The function "playSound" is defined. This function is executed when a key is pressed. It retrieves the audio element that corresponds to the pressed key code and its associated div element. If there is no audio element for the key, the function exits early. Otherwise, it adds the 'playing' class to the div element to show a visual effect and plays the audio element. The currentTime of the audio element is set to 0 to ensure the sound always starts playing from the beginning.

6. Next, a number of event listeners are added. The "transitionend" event is added to each key element (.key). When the transition of a key is completed, the "removeTransition" function is executed to remove the visual effects.

7. The "keydown" event is added to the window. When a key is pressed, the "playSound" function is executed to play the corresponding sound and show the visual effects.

What I have learned

From this challenge, I have learned how to structure consists of div elements representing different keys, each associated with a specific keyboard key and a corresponding sound.

Through JavaScript, I have implemented event listeners to detect key presses and trigger the playing of the associated sound and visual effect. I have also learned to handle the transitioned event to remove the visual effect class from the key element after the transition finishes. By understanding and applying these concepts, I have gained insights into event handling, audio playback, and creating interactive experiences in web development.