Lesson 03.08

Keyboard Events

- The pressing of a key is a **keydown** event.
- The release of a key is a **keyup** event.
- As with any event, a keyboard event can call a function.
- Keyboard events are listened for by the document.
- Syntax: **document.addEventListener('keyup', getKey)**
- The **event.key** property returns the name of the key
- If the key is the one we are listening for, run some function

1. Get the DOM elements: the container, the two boxes in the upper corners for displaying output, and the spaceship:

```
const container = document.querySelector('.container');
const keyBox = document.getElementById('key-box');
const pinBox = document.getElementById('pin-box');
const spaceShip = document.getElementById('space-ship');
```

2. Set the left position of the spaceship to equal half the window width, minus half the width of the spaceship. This puts the spaceship is the middle of the screen to start:

```
let leftPos = window.innerWidth / 2 - 128;
spaceShip.style.left = leftPos + 'px';
```

3. Set booleans to keep track of the font and dark mode states, which are toggled by pressing "f" and "d", respectively:

```
let serif = false;
let dark = false;
```

4. Set the speed of the spaceship. Each time the left or right arrow is pressed, the spaceship:

```
let speed = 20;
```

5. Have the document listen for the keyup event. On keyup, run a function that outputs the key and code to the keyBox. Since the event object is used by the function, pass the event object into the function as its argument. Since the function is so short, write it as an inline anonymous function, as opposed to an external, named function:

```
document.addEventListener('keyup', function(event) {
keyBox.innerHTML = `Key: ${event.key}<br>Code: ${event.code}`;
});
```

6. Have document listen for the keydown event. When the event takes place -- which is when ANY key is pressed -- call the onKeyPress function.

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

7. Define the onKeyPress function, passing in the event object:

```
function onKeyPress(event) {
```

8. Output the key and code that was pressed:

```
keyBox.innerHTML = `Key: ${event.key}<br>Code: ${event.code}`;
```

Check if the key is 'c', 'd', 'p', 'n' or the left or right arrow.

setting random background color

9. Check if the key is "c" for "color", or if the spacebar was pressed:

```
if(event.key == 'c' || event.code == 'Space') {

    // 10. Generate three integers in the 0-255 range
    let R = Math.floor(Math.random() * 255);
    let G = Math.floor(Math.random() * 255);
    let B = Math.floor(Math.random() * 255);
```

10. Concatenate the numbers into the rgb() method:

```
let randRGB = `rgb(${R}, ${G}, ${B})`;

// 12. Set the body background color to the random RGB color:
document.body.style.backgroundColor = randRGB;
```

Alternatively, we ca generate the random color as a hex value, in which case we only need one random number.

13. Comment out steps 10-12, and make a new random integer in the 0 - 16777215 range (256 x 256 x 256):

```
let r = Math.floor(Math.random() * (256 ** 3));

// 14. Convert the random number to a base-16 (0-9, A-F) hexadecimal
value by calling the toString(16) method on the number.
// let randHex = "#" + r.toString(16); // hexify the number
let randHex = getRandHexColor();

// 15. Set the body background color to the random hex color:
// document.body.style.backgroundColor = randHex;
document.body.style.backgroundColor = randHex;

// 16. Set the pinBox text and color to the random color:
// pinBox.style.color = randHex;
pinBox.style.color = randHex;
// pinBox.textContent = randHex;
pinBox.textContent = randHex;
```

Toggling between Dark Mode and Light Mode for the container

17. Check if the key is "d" for "dark":

```
} else if(event.key === 'd') {

    // 18. If dark boolean is currently false:
    if(!dark) {

    // 19. Switch to dark mode by adding and removing classes:
    container.classList.remove('light-mode');
    container.classList.add('dark-mode');

    // 20. Else, dark mode is already true
    } else {

    // 21. Switch to light mode:
    container.classList.remove('remove-mode');
    container.classList.add('light-mode');
    }

    // 22. Flip the dark boolean:
    dark = !dark;
```

18. Check if the key is "f" for "font":

```
} else if(event.key === 'f') {
```

```
    // 24. If serif boolean is currently false:
    if(!serif) {

    // 25. Set the body font family to 'serif':
    document.body.style.fontFamily = 'serif';

    // 26. Else, serif boolean is already true:
    } else {

    // 27. Set the body font family to 'sans-serif':
    document.body.style.fontFamily = 'sans-serif';
    }

    // 28. Flip the serif boolean:
    serif = !serif;
```

19. Check if the key is "p" for "pin" or "n" for "number":

```
} else if(event.key === 'p' || event.key === 'n') {

    // 30. Generate a random number in the 0-9999 range:
    let r = Math.floor(Math.random() * 10000);

    // All pin numbers need to be four digits, so we need to add
leading zero(es) to numbers in the 0-999 range.

    // 31. Declare a variable to store the pin as a string:
    let pin;

    // 32. If r is 0, set pin to be four zeroes:
    if(r == 0) {
    pin = "0000";

    // 33. else if r is less than 10, add three leading zeroes:
    } else if(r < 10) {
    pin = "000" + r;

    // 34. else if r is less than 100, add two leading zeroes:
    } else if(r < 100) {
    pin = "00" + r;

    // 35. else if r is less than 1000, add one leading zero:
    } else if(r < 1000) {
    pin = "0" + r;

    // 36. else r is a four-digit number, so use r as the pin:
    } else {
    pin = r;
    }

    // 37. Output the pin to the "pin box":
    pinBox.innerHTML = 'PIN:<br>' + pin;
```

20. If the Left Arrow was pressed:

```
```js
} else if(event.code == "ArrowLeft") {  // OR: event.key == "ArrowLeft"

    // 39. If the spaceship isn't already all the way left:
    if(leftPos > 0) {

    // 40. Reduce the leftPos value by the speed (default is 20):
    leftPos -= speed;

    // 41. Move the spaceship left by the speed value:
    spaceShip.style.left = leftPos + 'px';

    }
```
```

42. If the Right Arrow was pressed:

```
```js
} else if(event.code === "ArrowRight") { // OR: event.key == "ArrowRight"

    // 43. If the spaceship isn't already all the way to the right (250px
from the right end of the window):
    if(leftPos < window.innerWidth - 250) {

    // 44. Increase the leftPos value by the speed (default is 20):
    leftPos += speed; // subtract 10 from leftPos

    // 45. Move the spaceship right by the speed value:
    spaceShip.style.left = leftPos + 'px';
    }
```
```

46. If the key pressed is NOT 'c', 'd', 'p', 'n', left arrow or right arrow

```
```js
} else {

    // 47. Output a message to the pin box, so at least something happens:
    pinBox.textContent = "Nothing doing!";
    }
```
```

```
}
```

Random Hex Color

- A base 10 number uses the digits 0123456789 only
- A base-16 value uses the digits 0123456789ABCDEF only
- A hexadecimal color is a base 16 value
- The **toString()** method converts a number to a string
- **toString(16)** converts a base 10 number to a base 16 string
- There exist 16,777216 RGB colors (256 x 256 x 256)
- Call toString(16) on an int from 0-16777215 to get a hex color
- Put **'#'** before the number to complete the hex color

48. Declare a function that does ONE thing: makes a random hex color:

```
function getRandHexColor() {

    // 49. Generate a random 16-digit float from 0-16777215
    let r = Math.floor(Math.random() * 256 ** 3);

    // 50. Make a base-16 hexadecimal color from the random number and
return it
    let randHex = '#' + r.toString(16);

    // 51. return the hexadecimal color value
    return randHex;

}
```

Switch to using the **getRandHexColor()** function:

52. Comment out steps 10-13

53. In step 14, set **randHex = getRandHexColor()**

54. In steps 15 and 16, set all three values to **randHex**