Section-13 ⇒ Advanced JS and DOM Manipulation

① **Drum Kit** (what we will build) → Diff keys (∞ keys) tap produce different sounds of a drum.

② **Adding Event Listeners to a Button** → Buttons will let you know when user clicks on them → Event listener.

∴ first select button by,

⇒ `document.querySelector("button")`

Then type the method name,

⇒ `document.querySelector("button").addEventListener()` ★★

3 parameters added here

3 (Events parameters) ⇒ type → specify → "click" (listens to clicks).
listener → in this we call our fn
options.

∴ `document.querySelector("button").addEventListener("click", handleClick);`

Just name of the fn we call.

Note that we are not using () here

by If we do that ← it will call method not waiting for click.

for fn.

★★ Can study Anonymous fn if you want.

(3) We added EventListener to our one button previously, to add to all the buttons either we can use querySelectorAll("button")[1] (or) we can use loop.

↳ then 2
then 3 etc

» for (var i=0; i< document.querySelectorAll(".drum").length; i++) {

}

---

(4) Higher Order fns
Eg Passing fns as Arguments.

» we saw this → document ---- EventListener(input1, input2);

* input 1 → specifies what event to listen to
input 2 → what to do when event is detected.
↳ (called a callback fn. if fn used)

» Calling fns as arguments

» function multiply(num1, num2) {
return num1*num2; }

function add(num1, num2) {
return num1+num2; }

function calculator(num1, num2, operator) {
return operator(num1, num2) }.

Calling
» calculator(2, 3, add)
» 5.

Here calculator is called Higher order fn.

---

(5) How to Play Sounds on website.

- var audio = new Audio('audio_file.mp3'); (or address)
  audio.play();
- Has many properties inside like when to play, etc. in documentation.
- we also added bg images (using CSS) to each button.
- TO know which button is clicked we use [this].

Eg:1

console.log(this)
// Click on a button
output → console prints (button. ... some code).

If console.log(this.innerHTML).
// Click on button.
Output → a, w, etc only printed.

---

(6) Javascript Objects.

- A JavaS obj ⟶ var houseKeeper1 {
  yearsOfExp = 2;
  Name = x;
  glassesBroken = 3;
  }

Accessing → ("My Name is" + houseKeeper1.Name)

* Constructor func.

* func BellBoy (name, age, lang, clean) {
  this.name = name;
  this.age = age;
  this.lang = lang;
  this.clean = function() { } → Anonymous fn.
  }
  └ Anything you want.

[In constructor fn has starting var aph as capital].

calling & creating a Object. → var bellboy1 = new BellBoy ("Timmy", 19, "English");

* For sound we using (if & else) (or) (Switch).

---

(7) Using Keyboard Event Listners to check for key presses.

* Now we want our website to not only detect key presses but also detect keyword presses.

* So to implement it instead of listening for "click" we can listen for "keydown".

* ∴ document.addEventListener("keydown", fn() { });*

(8) Callbacks.

* document.addEventListner("keydown", fn(parameter)); → call back fn.

function fn(parameter) {
  // what it does
}

*Callback Letter than above fn..

(9) Adding Animation to the Website.

→ Remember for make sound and for animation (i.e flash, etc.) we have to include them in both when key is pressed by mouse (or) alphabet pressed in keyboard.

→
```
var active = document.querySelector("." + buttonpressed);
// Using this we select button pressed.
```

→ // Now just define a fn in CSS and call it in JS.

→
```
active.classList.add("pressed");
```
↳ classes in CSS ↳ fn defined in CSS for giving shadow (or) colour.
↳ described above. ↳ defined by us.

→ After this we use setTimeout (fn, millisec, param1, param2...)
↓ as many req.

∴ we defined another fn, so button gets back to it's original state after a timeout, looking like a flash.