


2022 퍼블리셔 부트캠프 011

- 예제: DrumKit(new Audio) // debugger Constructor Function // Higher order(Callback) Function

!Tips & Links

※ 강의 <https://www.udemy.com/course/the-complete-web-development-bootcamp/>

new Audio	audio=new Audio('dir'); audio.play(); // creating HTMLMediaElement (audio)
debugger	크롬, debugger; + 함수입력 // function call 순서대로 확인 
예제: 계산기	<pre> add=(num1, num2)=>num1+num2; //호이스팅xx subtract=(num1, num2)=>num1-num2; multiply=(num1, num2)=>num1*num2; division=(num1, num2)=>num1/num2; calculator=(num1, num2, operator)=>operator(num1, num2); function handle_promptInput(num1, num2, op){ let result="다시 입력해주세요"; op==0?result=calculator(num1, num2, add):""; op==1?result=calculator(num1, num2, subtract):""; op==2?result=calculator(num1, num2, multiply):""; op==3?result=calculator(num1, num2, division):""; return result; } //prompt값을 함수이름으로 못 읽어서 만들 </pre>
Higher Order Function & Use debugger	<pre> alert('계산기!'); alert(handle_promptInput(num1=Number(prompt("type first number:")), num2=Number(prompt("type second number:")), operator=prompt("select what to do\n [0]: add\n [1]: subtract\n [2]: multiply\n [3]: division\n"))) </pre>

※ Constructor
Function
(Initialise Object)
var ~ = new ~(.)

```

function FirstWordCapitalize( key1,key2,key3 ){
  this.key1 = value1;
  this.key2 = value2;
  this.key3 = value3;
  this.method1=function(){
    나는 메소드메소드;
  }
}

var object1 =
new FirstWordCapitalize( val1, val2, val3 );
var object2 =
new FirstWordCapitalize( val1, val2, val3 );

object1.key1 == val1
object1.method1(); == 메소드메소드

```

※ switch & case
if 쓸 때 switch 쓸 수
있나 한 번 생각해보기
활용 많이 하기
(까먹으니깐)

```

switch (~~) {
  case 'A' // ~~ == 'A'
    do_this; // no parenthesis, yes colon:
    break; // yes break;
  case 'B'
    do_this;
    break;
  default: log_something; // don't forget.
}

```

event-> e 또는 evt 씸 ~.addEventListener('~', functions(e))	higher order function callback function	able to take a function as an input function that gets passed in as an input allows as to wait until somethings finish happening
---	--	--

```

/* -----<lang: en>----- */
parentheses() << straight up function call ↔ passing in the name of the function as an input↵
anonymous function(){ // function name(1, 2, operator){ return operator(1, 2) } <<함수 조립
. <- dot notation // higher order function: able to take a function as an input

```