

```

previous = current;
current.classList.remove("transition-in");
current.classList.add("transition-out");

setTimeout(function(){
    current.classList.remove("transition-out");
}, 1000);

index = index - 1;
current = $list.children[index];
current.classList.add("transition-in");

console.log("previous", index, current);
}

};

$next.onclick= function () {
    if (index < limit) {
        $list.classList.remove("direction-left");
        $list.classList.add("direction-right");

        previous = current;
        current.classList.remove("transition-in");
        current.classList.add("transition-out");

        setTimeout(function(){
            previous.classList.remove("transition-out");
        }, 1000);

        index = index + 1;
        current = $list.children[index];
        current.classList.add("transition-in");
        console.log("next", index, index, current);
    }
};

```

```

.carousel-list{
    position: relative;
    margin: 0;
    padding-bottom: calc((800/1400)*100%);
    overflow: hidden;
    width: 100%;
    list-style: none;
}

.carousel-list .carousel-item.direction-right{
    transform: translateX(-100%);
    will-change: transform;
}

.carousel-list.direction-right .carousel-item.transition-in{
    transform: translateX(0);
    transition: transform 500ms;
}

.carousel-list.direction-right .carousel-item.transition-out{
    transform: translateX(100%);
    transition: transform 500ms;
}

.carousel-list .carousel-item.direction-left{
    transform: translateX(-100%);
    will-change: transform;
}

.carousel-list.direction-left .carousel-item.transition-in{
    transform: translateX(0);
    transition: transform 500ms;
}

.carousel-list.direction-left .carousel-item.transition-out{
    transform: translateX(100%);
    transition: transform 500ms;
}

.carousel-item{
    position: absolute;
    top: 0;
    left: 0;

    width: 100%;
    height: 100%;
}

```

- replaceAll() → minden előfordulást kicserél
 - toUpperCase() → konvertálás nagybetűre
 - toLowerCase() → konvertálás kisbetűre
 - concat() → két vagy több string összekapcsolása
 let text1 = "Hello"; let text2 = "World";
 let text3 = text1.concat(" ", text2);
 - trim() → eltávolítja a szóközöket a string kezdetéről és végéről
 - trimStart() → csak az elejétől töröl
 - trimEnd() → csak a végétől töröl
 - padStart() → padding a kezdeténél
 - padEnd() → padding a végénél
 let text = "5";
 let padded = text.padStart(4, "0");
 // 0005 (output)
 let text = "5";
 let padded = text.padEnd(4, "x");
 // 5xxx

Note: to pad a number convert the number to a string first

let numb = 5;
 let text = numb.toString();
 let padded = text.padEnd(4, "0");
 // 0005

- charCodeAt() → returns the unicode of the character at a specified index (an integer between 0 & 65535)
 - split() → convert a string to an array

String Search

• String.indexOf() → returns the index (position) the 1st occurrence of a str in a str
 let text = "Please locate where 'locate' occurs";
 let index = text.indexOf("locate");
 // output: 7
 • LastIndexOf() → same just with the last occurrence
 // output: 21
 • hasOwnProperty() → returns true if the property exists in the object
 • hasOwnProperty() → returns true if the property exists in the object
 • search() → returns the position of a match
 • match() → returns a boolean value
 • match() → returns a boolean value
 • match() → returns a boolean value