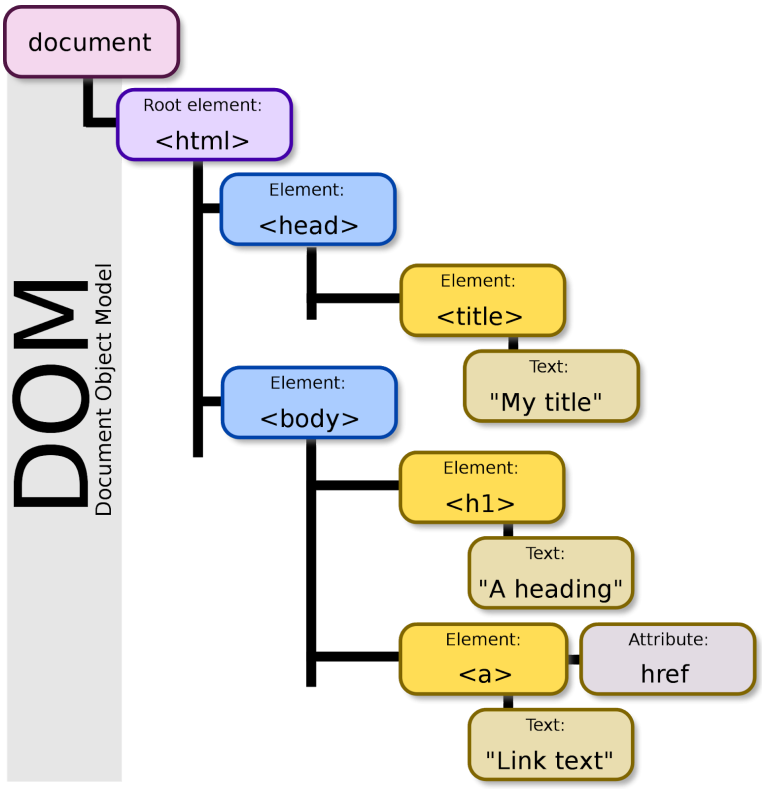
The **Document Object Model (DOM)** connects web pages to scripts or programming languages by representing the structure of a document—such as the HTML representing a web page—in memory.

The **DOM represents a document with a logical tree.** Each branch of the tree ends in a node, and each node contains objects. DOM methods allow programmatic access to the tree. With them, you can change the document's structure, style, or content.

https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model

JavaScript DOM Introduction to creating interactive and dynamic web pages.



How to write JavaScript editor used in the course and resources for lessons

Resources for writing code and for the upcoming lessons.

Code editor used in the course

https://code.visualstudio.com/

Browser Used in the course is Chrome - DevTools

Chrome DevTools is a set of web developer tools built directly into the Google Chrome browser. DevTools can help you edit pages on-the-fly and diagnose problems quickly, which ultimately helps you build better websites, faster.
https://developers.google.com/web/tools/chrome-devtools

Open DevTools

There are many ways to open DevTools, because different users want quick access to different parts of the DevTools UI.

When you want to work with the DOM or CSS, right-click an element on the page and select Inspect to jump into the Elements panel. Or press Command+Option+C (Mac) or Control+Shift+C (Windows, Linux, Chrome OS).

When you want to see logged messages or run JavaScript, press Command+Option+J (Mac) or Control+Shift+J (Windows, Linux, Chrome OS) to jump straight into the Console panel.

Lesson Challenge

1. Use QuerySelector to select page element
2. Overview of Editor and browser devtools
3. Select textContent of element
4. Update textContent of Element
5. Try multiple ways of element selection.

```
<!DOCTYPE html>
<html>
 <head><title>JavaScript DOM</title>
 <style>
   .myEle{
     color:red;
   }
 </style>
 </head>
 <body>
```

```
    <div id="myId" class="myEle">Hello 2</div>
    <script src="dom1.js"></script>
  </body>
</html>

const myEle1 = document.querySelector('.myEle');
const myEle2 = document.querySelector('#myId');
const myEle3 = document.querySelector('div');

myEle1.textContent = "hello world";
```

2 How to select all matching page elements with JavaScript QuerySelectorAll

DOM examples and how the DOM relates to JavaScript Code. Mini JavaScript Object with nested objects to illustrate a simple example of the DOM element tree. Update and selection of page elements using querySelector and querySelectorAll to select web page elements and update the value of the object. Select the element and manipulate the contents with textContent property value.

Lesson Challenge

1. Create a JavaScript object that simulates the DOM of page elements
2. Try console.dir(document) locate the elements within the object
3. Go to any web page - update the body contents with Hello World
4. Select and return multiple elements into a nodelist, update the count of elements along with the textContent of the element.

```
<!DOCTYPE html>
<html>
```

```
 <head><title>JavaScript DOM</title>
 <style>
   .myEle{
     color:red;
   }
 </style>
 </head>
 <body><div class="myEle">Hello 3</div><div class="myEle">Hello
3</div><div class="myEle">Hello 3</div><div class="myEle">Hello
3</div><div class="myEle">Hello 3</div><div class="myEle">Hello
3</div><div class="myEle">Hello 3</div><div class="myEle">Hello
3</div>
   <div id="myId" class="myEle">Hello 2</div>

   <script src="dom1.js"></script>
 </body>
</html>



const myEle1 = document.querySelector('.myEle');
const myObj = {"html":{"body":{"div":"Hello"}}}
console.dir(document);
myEle1.textContent = "hello world";

const eles = document.querySelectorAll('.myEle');
eles.forEach((el,index)=>{
   console.log(el.textContent);
```

```
    el.textContent = `Hello World ${index+1}`;
})
```

3 Web Page Element Style Attribute Update with JavaScript

Page element style values within style property of the element. Select an element and update the style values, get and set Attributes of the page element.

Lesson Challenge

1. Select the Element - update HTML of element with innerHTML property value
2. Update color and background color of selected element
3. Apply various styling values to the style attribute with JavaScript
4. List out available style properties and try

```
const output = document.querySelector('#myId');
const html = "<h1>Hello World</h1>";
output.textContent = html;
output.innerHTML = html;

output.style.color = 'red';
output.style.backgroundColor = 'yellow';
output.style.border = '5px solid black';
output.style.borderRadius = '25px';

const el = output.style;
console.log(el);
```

```
el.textAlign = 'center';
el.textTransform = 'uppercase';

el.margin = '25px';
```

4 Attributes ClassList Add Remove Toggle Contains within class of page element

Explore how to select and update the element attributes, add new attributes and get contents of existing attributes. Useful classlist methods to toggle existing class, add and remove classes and check if the class exists on an element returning boolean value.

Lesson Challenge

1. Select element using classList add a class to the element
2. remove a class from the element
3. Toggle a class on the element
4. Get Attribute from element and Set Attribute for element
5. Get id with dot notation and set id with dot notation
6. Update all hyperlinks to add target blank attribute
7. Check if the element contains a class.

```
const ele1 = document.querySelector('.myEle');
const btn = document.querySelector('button');
const myLinks = document.querySelectorAll('a');

console.log(ele1.classList);

/* ele1.classList.add('red');
```

```
ele1.classList.toggle('myEle');
ele1.classList.remove('red');
ele1.classList.toggle('blue'); */

console.log(ele1.getAttribute('class'));
ele1.setAttribute('class','red upper');
ele1.setAttribute('id','newID');
console.log(btn.getAttribute('disabled'));
btn.setAttribute('disabled',true);

console.log(ele1.id);
ele1.id = 'anotherValue';

myLinks.forEach((myLink)=>{
   //myLink.innerHTML += '<br>';
   if(myLink.classList.contains('red')){
       myLink.textContent += ' RED';
   }
   myLink.innerHTML += '<br>';
   myLink.setAttribute('target','_blank');
})



<!DOCTYPE html>
<html>

<head>
```

```
 <title>JavaScript DOM</title>
 <style>
   .myEle {}

   .upper {
     text-transform: uppercase;
   }

   .red {
     color: red;
   }

   .blue {
     color: blue;
   }
 </style>
</head>

<body>
 <div id="myId" class="myEle">Hello 2</div>
 <button>Click Me</button>
 <a href="#" class="red">Click Me</a>
 <a href="#">Click Me</a>
 <a href="#">Click Me</a>
 <a href="#" class="red">Click Me</a>
 <a href="#">Click Me</a>
 <a href="#">Click Me</a>
```

```
 <script src="dom3.js"></script>
</body>


</html>
```

5 Add HTML to Page with JavaScript Code innerHTML property of web page elements.

Select a page element with JavaScript - create page elements with loop from JavaScript to create multiple elements on the page. Add HTML to the page elements with innerHTML property value. Setting hyperlink attribute to have target blank, selecting all hyperlinks on page. Creating images with img elements as html code for page. Generate a random color with JavaScript string method. Lesson also includes a challenge to add html to a parent element and then select the new elements with JavaScript.

Lesson Challenge

1. Select element to update
2. Create loop adding to the HTML of hyperlinks
3. Use querySelectorAll to select page hyperlinks and add target attribute
4. Create images tags with JavaScript - Select all images on the page and if the src if null then add images to the src path
5. Generate random colors for the images.
6. **Challenge : Lesson challenge to add html to parent element**

```


<!DOCTYPE html>
<html>
```

```
<head>
 <title>JavaScript DOM</title>
 <style>
   .myEle {}

   .upper {
     text-transform: uppercase;
   }

   .red {
     color: red;
   }

   .blue {
     color: blue;
   }
 </style>
</head>

<body>
 <div id="myId" class="myEle">Hello 2</div>
 <button>Click Me</button>

 <script src="dom4.js"></script>
</body>

</html>
```

```
const main = document.querySelector('#myId');
main.innerHTML = '';
//https://dummyimage.com/200x100/ffffff/000000&text=dummyimage.com+rocks! //background /color
const url =  '//dummyimage.com/200x100/';
for(let i=0;i<40;i++){
   main.innerHTML += `<img>`;
}

const imgs = document.querySelectorAll('img');
imgs.forEach((el,index)=>{
   if(!el.getAttribute('src')){
       const myImg = url + `${myRandomColors()}/${myRandomColors()}&text=New Image ${index+1}` ;
       el.setAttribute('src',myImg);
   }
})

function myRandomColors(){
   return Math.random().toString(16).substr(2,6);
}

/* for(let x=0;x<10;x++){
   main.innerHTML += `<a href="#">Click Me ${x+1}</a><br>`;
}

const eles = document.querySelectorAll('a[href="#"]');
```

```
console.log(eles);
eles.forEach((el,index)=>{
   console.log(el);
   if(index %3 == 0){
       el.classList.add('blue');
   }
   if(index %2 == 0){
       el.classList.add('red');
   }
   el.setAttribute('target','_blank');
})

*/
```

6 JavaScript to Create new page elements and Remove Elements

Use of createElement method to generate new page elements with JavaScript Code. Append Prepend AppendChild to parent Element methods of adding page elements to page. JavaScript insertBefore to add within an element and get the callback value. Coding Challenge to create multiple image elements adding images and properties with JavaScript Code.

Lesson Challenge

1. Create elements using createElement method
2. Append Prepend AppendChild to parent Element
3. Try insertBefore to add within an element and get the callback value.

4. **Challenge** create multiple images adding them to the page select img objects update the properties.

```
<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
   .red{
     border: red solid 5px;
   }
   .box{
     border: black solid 5px;
   }
 </style>
</head>

<body>
 <div class="output"></div>
 <button>Click Me</button>

 <script src="dom5.js"></script>
</body>

</html>
```

```
const output = document.querySelector('.output');
const url =  '//dummyimage.com/200x100/';

for(let i=0;i<20;i++){
   const img = document.createElement('img');
   const myImg = url +
`${myRandomColors()}/${myRandomColors()}&text=New Image ${i+1}`
;
   img.setAttribute('src',myImg);
   img.classList.add('box');
   if(i%3 ==0){
       img.classList.add('red');
       img.classList.remove('box');
   }
   //img.style.border = `${i+1}px solid red`;

   output.append(img);

}
function myRandomColors(){
   return Math.random().toString(16).substr(2,6);
}
/* const el1 = document.createElement('div');
const el2 = document.createElement('h1');
el2.textContent = 'Hello World';
const el3 = document.createElement('h2');
el3.textContent = 'Prepend Element';
const el4 = document.createElement('h2');
```

```
el4.textContent = 'Append Element';

const el5 = document.createElement('h2');
el5.textContent = 'Five';
const el6 = document.createElement('h2');
el6.textContent = 'Six';

el1.append(el2);
el1.append(el4);
el1.prepend(el3);
console.log(el1);
//el1.innerHTML = '<h1>Hello World</h1>';
output.append(el1);
el2.style.backgroundColor = 'red';

document.body.append(el2);

const val1 = output.append(el5);
const val2 = output.appendChild(el6);
const val3 = output.insertBefore(el5,el6);

const myText1 = document.createTextNode('Hello World');
console.log(myText1);

val2.appendChild(myText1); */
```

7 JavaScript Traversing the DOM parents siblings children of elements.

Navigate the DOM tree, select a starting element and move to its related elements with JavaScript Code. Select element parent object, get list of elements children and child nodes. Select an element get the first last and siblings related to the current element. Move to the next element and update the element.

Lesson Challenge

1. Select element to start with
2. Get the element Parent element
3. Get the element children
4. Get the element siblings
5. Update related element to the selected one

```
const output = document.querySelector('.output');
const ulList = document.querySelector('ul');

for (let i = 0; i < 10; i++) {
   const li = document.createElement('li');
   li.textContent = `${i+4}`;
   ulList.append(li);
}

console.log(ulList.childNodes);
console.log(ulList.children);

ulList.childNodes.forEach((val) => {
```

```
    //console.log(val);

})

for (let i = 0; i < ulList.children.length; i++) {

    const el = ulList.children[i];
    console.log(el.previousElementSibling);
    const prev = el.previousElementSibling;
    if (prev != null) {
        console.dir(el.parentElement);
        prev.style.color = 'red';
        prev.textContent += `<br>Parent :
${el.parentElement.tagName} <br>ParentNode :
${el.parentNode.tagName}`
    }
}

<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
    .red {
      border: red solid 5px;
    }
```

```
   .box {
     border: black solid 5px;
   }
 </style>
</head>

<body>
 <div class="output">
   <ul>
     <li>One</li>
     <li>Two</li>
     <li>Three</li>
   </ul>
 </div>
 <button>Click Me</button>

 <script src="dom6.js"></script>
</body>

</html>
```

8 Click Events and Event Listeners with JavaScript

Create interactive page elements that can be clicked to run blocks of code. User actions to trigger code blocks with JavaScript. How to setup click only once,

addEventListeners and removeEventListeners. Create custom object property values. Update elements dynamically with code.

Lesson Challenge

1. Adding Event Listener for clicks on elements
2. How to add eventlisteners to an element and list of elements
3. Allow for only one click on the element
4. Add and remove click events
5. Set object custom value like counter
6. Customize properties of element on click
7. Dynamically add elements that are clickable to page.

```
<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
   .red {
     border: red solid 5px;
   }
li{
 padding:10px;
 margin:5px;
}
   .box {
     border: black solid 5px;
   }
   .counter{
```

```
      padding:5px;
      background-color: black;
      color:white;
    }
  </style>
</head>

<body>
  <div class="output">
    <ul>
      <li>One</li>
      <li>Two</li>
      <li>Three</li>
    </ul>
  </div>
  <button>Click Me</button>

  <script src="dom7.js"></script>
</body>

</html>

const output = document.querySelector('.output');
const btn = document.querySelector('button');

btn.addEventListener('click',addListItem);
```

```
const ul = document.querySelector('ul');
const eles = ul.querySelectorAll('li');
let counter = 0;
eles.forEach((li)=>{
   adder(li);
   //li.addEventListener('click',updateEle,{once:true});

})

function adder(li){
   counter++;
   li.counter = 0;
   const span = document.createElement('span');
   span.textContent = li.counter;
   span.classList.add('counter');
   li.textContent = `New List Item `;
   li.append(span);
   li.addEventListener('click',(e)=>{
       updateEle(li);
   });

}




function addListItem(){
   const li = document.createElement('li');
```

```
    adder(li);

    ul.append(li);
}




function updateEle(el){
    //console.log(e.target.counter);
    //const el = e.target;
    const span = el.querySelector('.counter');
    el.counter++;
    span.textContent = el.counter;
    console.log(el.children);
    if(el.classList.contains('red')){
        el.classList.add('box');
    }
    el.classList.toggle('red');

    //e.target.removeEventListener('click',updateEle);
}


/*
ul.addEventListener('click',(e)=>{
    console.log(e.target);
    const el = e.target;
```

```
    console.log(el.textContent);
    if(el.textContent =="One"){
        el.classList.toggle('red');
    }else if(el.textContent =="Two"){
        el.classList.toggle('box');
    }else{
        el.classList.toggle('box');
    }
})  */
```

9 How to add Mouse Event Listeners with JavaScript code.

Common mouse events to page elements, on mouse over movement and other actions. Create events with mouse actions and how to track the events, what the difference is between mouse over and mouse out.

Lesson Challenge

1. Add Mouse event on element.
2. Run code when mouse move, out, enter and over events occurs
3. Update element properties on event

```
const output = document.querySelector('.output');
const game =
{'wheel':0,'mouseover':0,'mouseenter':0,'mouseout':0,'mousemove'
:0,};

for(const prop in game){
```

```
    console.log(prop ,game[prop]);
    const el = document.createElement('div');
    game[prop] = {val:0,el:el};
    const span = document.createElement('span');
    el.textContent = prop + ': ';
    span.textContent = 0;
    output.append(el);
    el.append(span);
}

const val1 = document.querySelector('.test');

output.addEventListener('mouseover',(e)=>{
    updateProp(e.type);
    //output.style.backgroundColor='red';
})
output.addEventListener('mouseenter',(e)=>{
    updateProp(e.type);
    //output.style.backgroundColor='blue';
})
output.addEventListener('mouseout',(e)=>{
    updateProp(e.type);
    //output.style.backgroundColor='white';
})
output.addEventListener('mousemove',(e)=>{
    updateProp(e.type);
    //output.style.backgroundColor='white';
})
```

```
output.addEventListener('wheel',(e)=>{
   updateProp(e.type);
   //output.style.backgroundColor='white';
})

function updateProp(ev){
   console.log(game[ev]);
   game[ev].val += 1;
   game[ev].el.textContent = ev + ' : ' + game[ev].val;
}

<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
   .red {
     border: red solid 5px;
   }
   li{
     padding:5px;

   }
   .box {
     border: black solid 5px;
   }
   .counter{
```

```
    padding:5px;
    background-color: black;
    color:white;
  }
  .output{
    border: 1px solid black;
    width:200px;
    height:400px;
  }
 </style>
</head>

<body>
 <div class="output">
  <ul>
    <li>One</li>
    <li>Two</li>
    <li class="test">Three</li>
  </ul>
 </div>
 <button>Click Me</button>
 <script src="dom8.js"></script>
</body>
</html>
```

10 Events Listeners Keyboard Events with JavaScript Code

Track keyboard events, get key values and how events can be attached to input fields. Focus and Blur on input fields running events and JavaScript Code. Keyup and KeyDown events tracking arrow presses on keyboard.

Lesson Challenge

1. Create input fields
2. Add focus blur and change events to input elements
3. Listen for press of enter key
4. Key up and down on element and on document.
5. Update page elements on key press.

```
<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
.box{
 display: block;
 width:50%;
 margin:auto;
}
 </style>
</head>
```

```
<body>
 <div class="output">
 </div>
 <button>Click Me</button>
 <script src="dom9.js"></script>
</body>
</html>
const output = document.querySelector('.output');
const h1 = document.createElement('h1');
output.append(h1);
document.addEventListener('keydown',(e)=>{
   h1.innerHTML += e.key;
})
document.addEventListener('keyup',(e)=>{
   console.log(e.key);
})

for(let i=0;i<5;i++){
   const ele = document.createElement('input');
   output.append(ele);
   ele.classList.add('box');
   ele.setAttribute('type','text');
   ele.setAttribute('name','input'+i);
   ele.addEventListener('focus',(e)=>{
       ele.style.backgroundColor = 'red';
       ele.style.color = 'white';
```

```
        ele.style.border = '1px solid black';
    })
    ele.addEventListener('blur',(e)=>{
        ele.style.backgroundColor = 'white';
        ele.style.color = 'black';
    })
    ele.addEventListener('change',(e)=>{

        ele.style.border = '1px solid red';
    })
    ele.addEventListener('keyup',(e)=>{
        console.log(e.key);
        if(e.key =='Enter'){
            console.log(ele.value);

        }
    })
}
```

11 How to Move an Element with Arrow presses on Keyboard using JavaScript

Coding exercise with JavaScript and keyboard events. Track arrow key presses and update the element position on the page. Key press to move element on screen using JavaScript.

Lesson Challenge

1. Create an Element
2. Add Event Listener for keyboard event
3. Check the key with a condition and update x,y values if arrow keys are pressed
4. Create a new element when click that will move

```
const output = document.querySelector('.output');
const game = {x:0,y:0,speed:10,ele:null}
const keyz =
{ArrowRight:false,ArrowLeft:false,ArrowUp:false,ArrowDown:fa
lse};


document.addEventListener('keydown',(e)=>{
   console.log(e.key);
   if(!game.ele){
       game.ele = maker();
   }
   if(e.key == 'ArrowRight'){
       game.x+=game.speed;
```

```
        updatePostion();
    }
    if(e.key == 'ArrowLeft'){
        game.x-=game.speed;
        updatePostion();
    }
    if(e.key == 'ArrowUp'){
        game.y-=game.speed;
        updatePostion();
    }
    if(e.key == 'ArrowDown'){
        game.y+=game.speed;
        updatePostion();
    }
})
document.addEventListener('keyup',(e)=>{
    console.log(e.key);
})

function updatePostion(){
    game.ele.style.left = game.x + 'px';
    game.ele.style.top = game.y + 'px';
}


function maker(){
```

```
    const el = document.createElement('div');
    output.append(el);
    el.style.left = game.x + 'px';
    el.style.top = game.y + 'px';
    el.classList.add('box');
    el.style.backgroundColor =
'#'+Math.random().toString(16).substr(2,6);
    el.addEventListener('click',(e)=>{
        game.ele = maker();

    },{once:true})
    return el;
}




<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
 .box{
    position: absolute;
    left:0;
    top:0;
```

```
   width:30px;
   height:30px;
   border: 1px solid black;
   border-radius: 50%;
 }
 </style>
</head>

<body>
 <div class="output">
 </div>
 <script src="dom10.js"></script>
</body>
</html>
```

12 How to create smooth movement of page elements Animation Frame

Update to coding challenge to move page elements, adding animation frame for smooth motion of page elements on keyboard arrow clicks. Press arrow keys and move the element up down left and right with JavaScript Code.

Lesson Challenge

1. Adding RequestAnimationFrame for smooth animation
2. Use of getBoundingClientRect to get boundaries of element rectangle.
3. Update of element position if keys are pressed.

```
<!DOCTYPE html>
<html>

<head>
 <title>JavaScript DOM</title>
 <style>
 .box{
   position: absolute;
   left:0;
   top:0;
   width:30px;
   height:30px;
   border: 1px solid black;
   border-radius: 50%;
 }
 .output{
   width:400px;
   height:400px;
   border: 1px solid red;
 }
 </style>
</head>

<body>
 <div class="output">
 </div>
```

```
 <script src="dom10.js"></script>
</body>
</html>



const output = document.querySelector('.output');
const game = {x:50,y:50,speed:5,ele:{}}
game.ele = maker();
const keyz =
{ArrowRight:false,ArrowLeft:false,ArrowUp:false,ArrowDown:false}
;

let move = window.requestAnimationFrame(updatePostion);
document.addEventListener('keydown',(e)=>{
   if(e.code in keyz){keyz[e.code] = true;}
})
document.addEventListener('keyup',(e)=>{
   if(e.code in keyz){keyz[e.code] = false;}
})

function updatePostion(){
   const domRect = output.getBoundingClientRect();
   console.log(domRect);
   if(keyz.ArrowRight && game.x < (domRect.right -30) ){
       game.x += game.speed;
   }
   if(keyz.ArrowLeft && game.x > domRect.left ){
       game.x -= game.speed;
```

```
    }
    if(keyz.ArrowUp && game.y > domRect.top ){
        game.y -= game.speed;
    }
    if(keyz.ArrowDown && game.y < (domRect.bottom -30) ){
        game.y += game.speed;
    }
    game.ele.style.left = game.x + 'px';
    game.ele.style.top = game.y + 'px';
    move = window.requestAnimationFrame(updatePostion);
}


function maker(){
    const el = document.createElement('div');
    output.append(el);
    el.style.left = game.x + 'px';
    el.style.top = game.y + 'px';
    el.classList.add('box');
    el.style.backgroundColor =
'#'+Math.random().toString(16).substr(2,6);
    el.addEventListener('click',(e)=>{
        game.ele = maker();

    },{once:true})
    return el;
}
```