CM1040 Web Development Week 16 Lecture Note

Notebook:	Web Development		
Created:	2020-10-13 4:07 PM	**Updated:**	2020-12-15 5:03 PM
Author:	SUKHJIT MANN		

Cornell Notes	**Topic:** React to the User continued	Course: BSc Computer Science Class: Web Development CM1040[Lecture] Date: December 15, 2020

Essential Question:

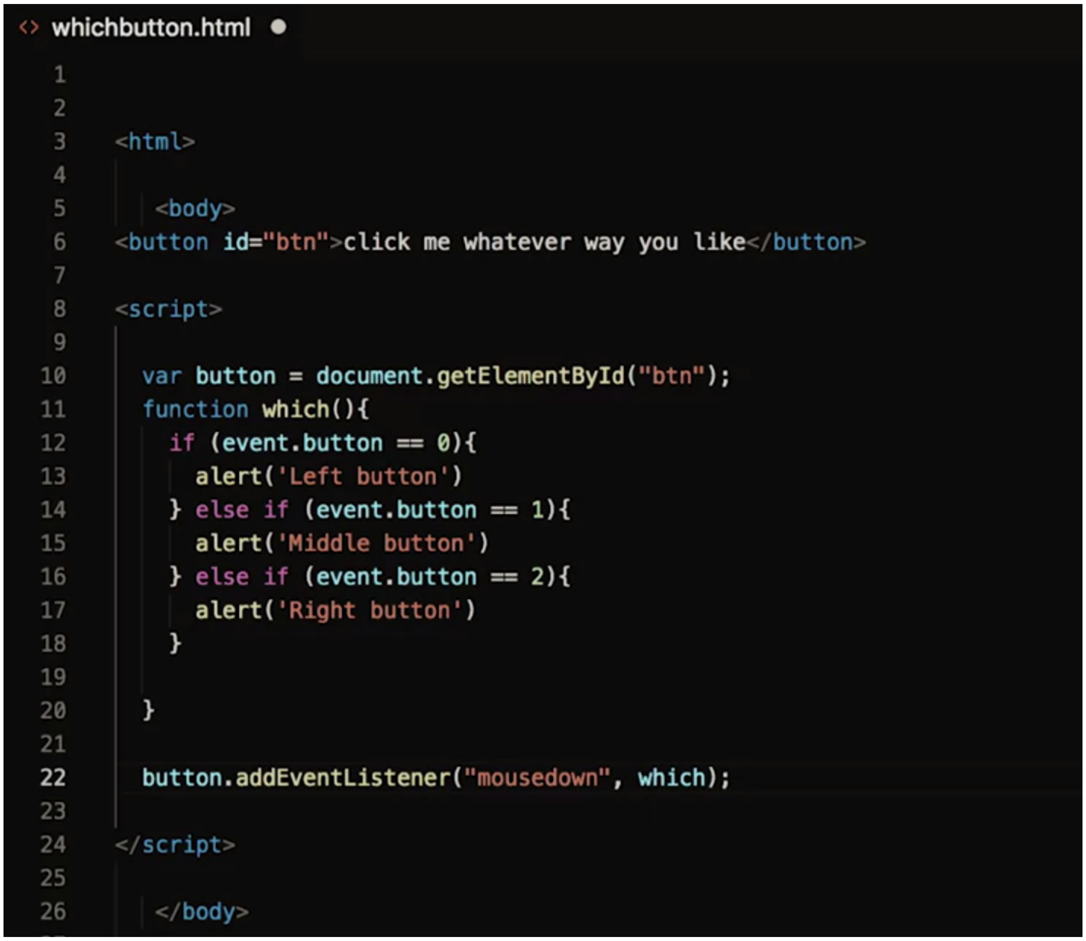
How does JavaScript reacts to User Input?

Questions/Cues:

- What are mouse-based events?
- What is the Click event?
- What are EventObjects?
- What is MouseMove event?
- What are the MouseDown/MouseUp events?
- What are the MouseOver and MouseOut events?
- What are focus/blur events?
- What are KeyPress/KeyDown/KeyUp Events
- What are TouchStart/TouchEnd events?

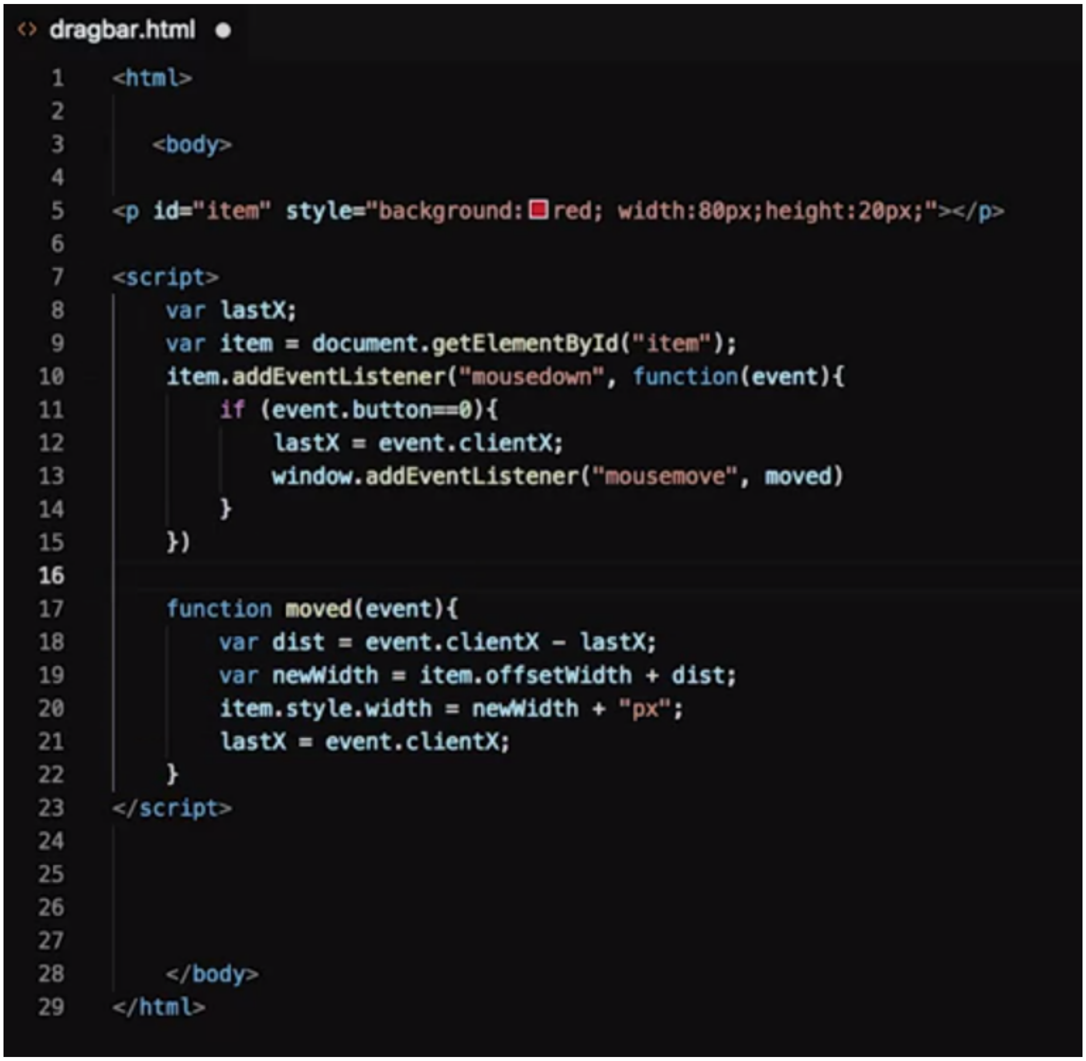
Notes

- Mouse-based events = onclick, click, MouseMove, Hover or MouseOver, and MouseOut are all mouse-based events or MouseEvents
- Click = for the click event to work we need be sure that the element we are actually binding a ClickEvent to is actually a clickable element
    - To if an element is clickable, we first create in HTML and then if we press tab on our keyboard and the element is highlighted then that means it's clickable
- EventObjects = variables that are specific to EventListeners. When an event that is attached to the DOM is triggered, all the relevant information about that event is gathered, and stored on an object called event. This object contains basic information such the element that caused the event, the type of event that occurred, and any other data that may be relevant to the particular event



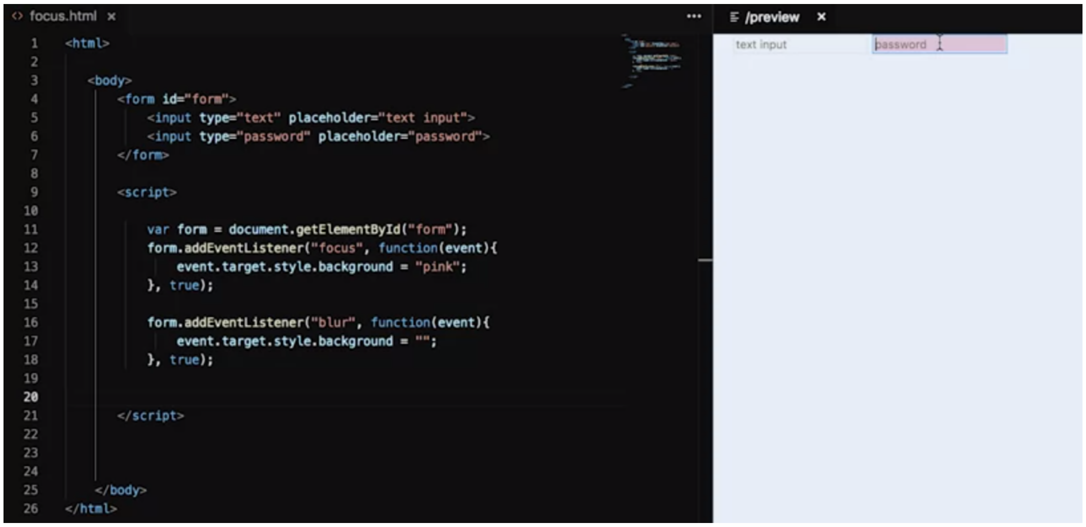
```


<html>

   <body>
<button id="btn">click me whatever way you like</button>

<script>

  var button = document.getElementById("btn");
  function which(){
    if (event.button == 0){
      alert('Left button')
    } else if (event.button == 1){
      alert('Middle button')
    } else if (event.button == 2){
      alert('Right button')
    }

  }

  button.addEventListener("mousedown", which);

</script>

   </body>
```

- Every time the user moves the mouse or pointer, the browser triggers a mouse move event



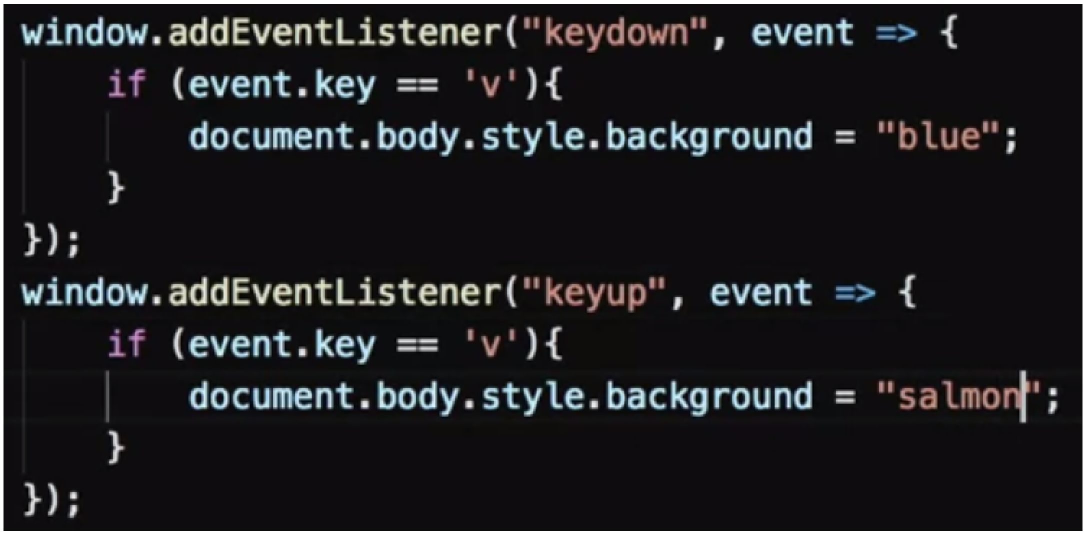
```
<html>

   <body>

<p id="item" style="background:red; width:80px;height:20px;"></p>

<script>
    var lastX;
    var item = document.getElementById("item");
    item.addEventListener("mousedown", function(event){
        if (event.button==0){
            lastX = event.clientX;
            window.addEventListener("mousemove", moved)
        }
    })

    function moved(event){
        var dist = event.clientX - lastX;
        var newWidth = item.offsetWidth + dist;
        item.style.width = newWidth + "px";
        lastX = event.clientX;
    }
</script>




    </body>
</html>
```

- The MouseDown event is triggered by the browser when any of the mouse buttons are pressed, doesn't matter which one
    - When user releases the mouse button another event is triggered called the MouseUp event. MouseUp is triggered even when a user presses a mouse button on an element and then moves away from that element. It's tied to MouseDown event and not the element.
        - The click event on the other hand is only triggered when the left mouse button is pressed and released on the same element
- MouseOver and MouseOut are referred to as hovering events
    - The MouseOver is triggered when the user positions their mouse pointer or cursor over an element
        - The MouseOut is triggered in the opposite case when the user is on the element and wants to move away from it
    - MouseOver and MouseOut are both linked with each other. So the MouseOut is always related to the same element as the MouseOver. It's a combination of the two that creates a hover effect
- When we navigate to a search form either by clicking on it or tabbing to it, a focus event is triggered by the browser. In this sense, when we click away or tab away from a search form then we are deactivating it and that is when the blur event is triggered by the browser
    - Focus is the click and blur is the unclick
- Just as with click, we need to check if an element is focusable before we attach a focus or blur event to it. A general rule is that link and form elements are focusable, this includes text inputs, check boxes and buttons

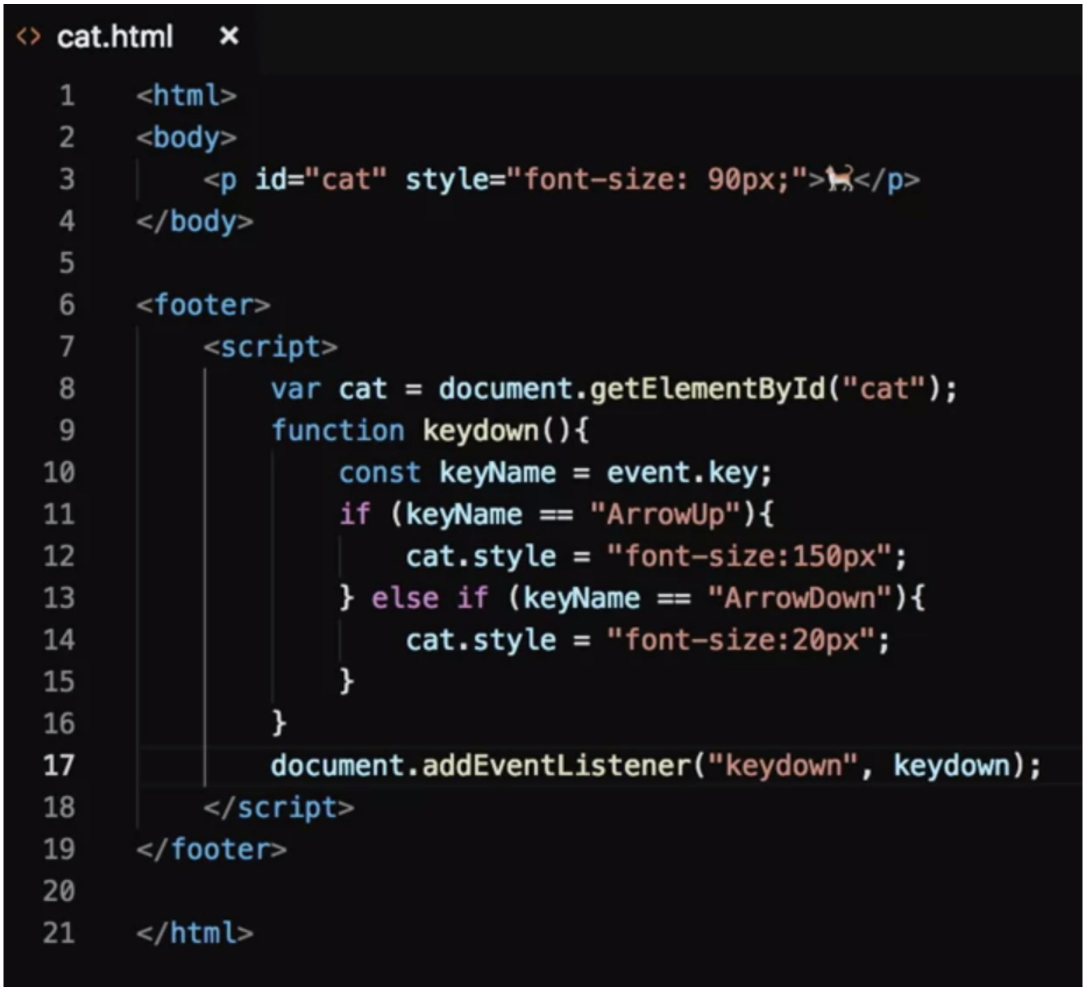


```
<html>

   <body>
       <form id="form">
           <input type="text" placeholder="text input">
           <input type="password" placeholder="password">
       </form>

       <script>

           var form = document.getElementById("form");
           form.addEventListener("focus", function(event){
               event.target.style.background = "pink";
           }, true);

           form.addEventListener("blur", function(event){
               event.target.style.background = "";
           }, true);


       </script>



    </body>
</html>
```

- ○ KeyDown/KeyUp are similar to the MouseDown and MouseUp where as they are refering to a user pressing a key and then lifting it. KeyPress on the other hand is similar to Click.



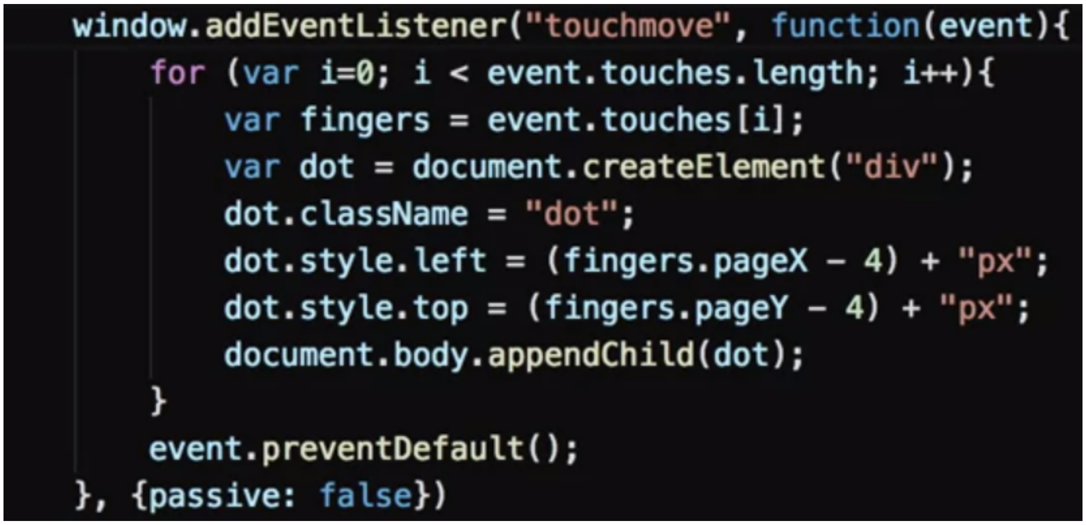
```
window.addEventListener("keydown", event => {
    if (event.key == 'v'){
        document.body.style.background = "blue";
    }
});
window.addEventListener("keyup", event => {
    if (event.key == 'v'){
        document.body.style.background = "salmon";
    }
});
```



cat.html

```html
<html>
<body>
    <p id="cat" style="font-size: 90px;">🐈</p>
</body>

<footer>
    <script>
        var cat = document.getElementById("cat");
        function keydown(){
            const keyName = event.key;
            if (keyName == "ArrowUp"){
                cat.style = "font-size:150px";
            } else if (keyName == "ArrowDown"){
                cat.style = "font-size:20px";
            }
        }
        document.addEventListener("keydown", keydown);
    </script>
</footer>

</html>
```

- touchStart is triggered when the user touches an element on the screen, and touchEnd is fired when the user removes their finger from that same element. When a finger or nose is moved, a touchMove is fired
- Since touchscreens can support multiple fingers on the screen unlike MouseMove, we get multiple coordinate points (x, y) called Event coordinates which track the position of the touch. This is why the touch event object has a touches property that we can look at. This property holds an array-like object that holds all the X, Y positions for all the touches occurring at the same time



```javascript
window.addEventListener("touchmove", function(event){
    for (var i=0; i < event.touches.length; i++){
        var fingers = event.touches[i];
        var dot = document.createElement("div");
        dot.className = "dot";
        dot.style.left = (fingers.pageX - 4) + "px";
        dot.style.top = (fingers.pageY - 4) + "px";
        document.body.appendChild(dot);
    }
    event.preventDefault();
}, {passive: false})
```

Summary

In this week, we learned about MouseEvents, KeyEvents and TouchEvents.