ANIMATING CONTENT

ANIMATING CONTENT

- fade elements in and out
- give elements a swipe animation
- animate them to move around the page

PARTS OF EVERY ANIMATION

- the starting state
- the movement toward the final goal
- the end state; stopping the animation

POSITIONING AND MOVING CONTENT

- In addition to changing the styling, we can also move it!

```
var divAdvert = document.getElementById("divAdvert");
divAdvert.style.position = "absolute";
divAdvert.style.left = "100px"; // set the left position
divAdvert.style.top = "100px"; // set the top position
```

TIMERS

- `setTimeout()` - one-shot timer
- `setInterval()` - continually firing timer

ONE-SHOT TIMER

```
var timerId = setTimeout(yourFunction, millisecondsDelay);
```

SETTIMEOUT

```
function doThisLater() {
    alert("Time's up!");
}

setTimeout(doThisLater, 3000);
```

STOP A TIMER

```
function doThisLater() {
    alert("Time's up!");
}

var timerId = setTimeout(doThisLater, 3000);

clearTimeout(timerId);
```

INTERVALS

```
var myTimerID = setInterval(myFunction, 5000);

clearInterval(myTimerID);
```

ACTIVITY: MAKE A CLOCK

- Create a simple HTML page
- Create a function that displays the current date and time.
- Use setInterval() to call the function every second.

ANIMATION EXAMPLE

Link

ANIMATION EXAMPLE

```
<body>
    <img id="cat" style="position:absolute; left: 0;" src="img/cat.gif">
    <!-- ...more code -->
```

Link

ANIMATION EXAMPLE

```
<script>
    var walkForwards = true;

    function catWalk() {
        var img = document.getElementById('cat');
        var currentLeft = parseInt(img.style.left);

        if (walkForwards && (currentLeft > (window.innerWidth-img.width))) {
            walkForwards = false;
        }

        if (!walkForwards && (currentLeft <= 0)) {
            walkForwards = true;
        }

        if (walkForwards) {
            img.style.left = (currentLeft + 10) + 'px';
        } else {
            img.style.left = (currentLeft - 10) + 'px';
        }
    }

    setInterval(catWalk, 50);
</script>
```

Link

ACTIVITY: CAT WALK

- Modify the code so that the amount of pixels moved in either direction is controlled by a global variable.
- Call it direction.
- Remove the walkForward variable.
- Change the code to use the new direction variable to determine when the animation should change directions.

Bonus: Move the cat up and down the page too!

Link

EVENTS

EVENTS

An **event** is an object that is sent when actions take place on your webpage, most often when a user interacts with your webpage.

For example, JavaScript creates an event when a user clicks an element.

```
element.addEventListener('click', function(event) {
  // code to be executed when user clicks
});
```

TYPES OF EVENTS

There are a variety of events. Some of the more common events are:

- **click:** Occurs when the user clicks on an element
- **mouseover:** Occurs when the pointer is moved onto an element
- **mouseout:** Occurs when the pointer is moved off an element
- **keyup:** Occurs when the user releases a key
- **load:** Occurs when a document has been loaded
- **focus:** Occurs when an element gets focus
- **blur:** Occurs when an element loses focus

CALLING FUNCTIONS FROM HTML

You can call a function directly from your HTML code:

```
<button id="myBtn" onclick="sayHi()">Click Me!</button>
```

```
function sayHi (event) {
  alert('Hi!');
};
```

CALLING FUNCTIONS FROM JAVASCRIPT

You can call a function from the addEventListener:

```
<button id="myBtn">Click Me!</button>
```

```
var button = document.getElementById("myBtn");

button.addEventListener("click", function (event) {
  alert("Hi!");
});
```

or

```
var button = document.getElementById("myBtn");

var sayHi = function (event) {
  alert("Hi!");
};

button.addEventListener("click", sayHi);
```

ACTIVITY: MOUSEOVER

- Go back to your simple HTML page.
- Make some JavaScript code fire after a mouseover event.

PREVENTING DEFAULTS

Elements like links and checkboxes have default behaviors determined by the browser. However, the `event` object has a built-in method to prevent the default behavior

Our anchor link in HTML

```
<a id="myLink" href="https://www.sait.ca/">SAIT</a>
```

Code to prevent going to link's href on click

```
var link = document.getElementById("myLink");

link.addEventListener("click", function(event) {
  event.preventDefault();
});
```

CURRENTTARGET

The event's `currentTarget` references the element the event listener was attached to.

Our button in HTML:

```
<button id="myBtn">Click Me!</a>
```

This code adds styles and text to our clicked button

```
myButton = document.getElementById("myBtn");

myButton.addEventListener("click", function(event) {
  btn = event.currentTarget;

  btn.style.backgroundColor = 'red';
  btn.innerHTML = 'Clicked!';
};
```

ACTIVITY: LINK ERROR

Write code that targets this link:

```
<a href="https://www.sait.ca/" id="saitLink">SAIT</a>
```

When a user clicks the link, the page should display an error message instead of going to the SAIT homepage.