Comparison Operators

Comparison operators help to make decisions that determine which lines of code should be run next.

Comparison operators allow you to compare values an test whether a condition is met or not.

They return single values of True or False.

Comparison Operators

>	Greater than	5 > 3 returns true
<	Les than	4 < 9 returns true
>=	Greater than or equal	7>=6 returns true
<=	Less than or equal	8<=8 returns true
!=	Is not equal to	‘Hello’ != ‘Goodbye’ returns true
==	Is equal to	‘Hello’ == ‘Goodbye’ returns false
===	Strict equal to	‘3’ ===3 returns false
!==	Strict not equal to	‘3’ !== 3 returns true

Comparison Operators

(score >= pass)

operand operand

comparison
operator

Comparison Operators

(score >= pass) > (highScore1 + highScore2)

operand

operand

comparison operator

Logical Operators

Logical operators allow you to compare the results of more than one comparison operator.

&& Logical AND

|| Logical OR

! Logical NOT

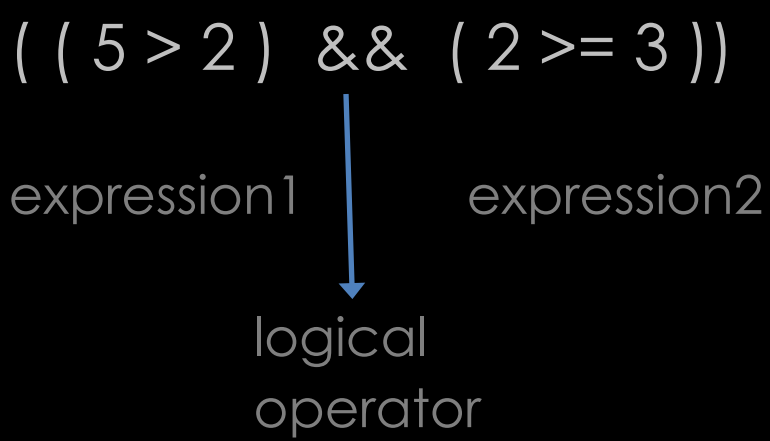
Logical Operators

((2<6) && (3>=2)) returns true

((2<6) || (2>1)) returns true

!(2 <1) returns true

Logical Operators



Logical Operators - example

```
var score1 = 8;
var score2 = 8;
var pass1 = 6;
var pass2 = 6;
var passBoth = (score1 >= pass1) && (score2 >= pass2);

var msg = 'Both rounds passed: ' + passBoth;
var el = document.getElementById('answer'); el.innerHTML = msg;
```

Logical Operators – example

```
var score1 = 8;
var score2 = 8;
var pass1 = 6;
var pass2 = 6;
var minPass = (score1 >= pass1) || (score2 >= pass2);

var msg = 'Resit required: ' + !(minPass);
var el = document.getElementById('answer'); el.innerHTML = msg;
```

If statement

The if statement evaluates a condition.

If the condition is true, all statements in the subsequent code block between opening curly brace and closing curly brace are executed.

```
if (condition) {
        block of code to be executed if the condition is true
}
```

If statement

```
if  (score >= 50)  {
        congratulate();
}
```

If … else statements

The if …else statement evaluates a condition.

If the condition is true, all statements in the first code block are executed.

If the condition is false, the second code block is executed instead.

```
if (condition) {
        block of code to be executed if the condition is true
} else {
        block of code to be executed if the condition is false
}
```

If … else statements

```
if  (score >= 50)  {
        congratulate();
}
else {
        encourage();
}
```

If … else statements

If the time is less than 20:00, create a "Good day" greeting, otherwise "Good evening":

```
if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

If … else statements – good morning example

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to display a time-based greeting:</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
```

If … else statements – good morning example

```
<script>
function myFunction() {
    var greeting;
    if (new Date().getHours() < 20) {
        greeting = "Good day";
    } else {
        greeting = "Good evening";
    }
    document.getElementById("demo").innerHTML = greeting;
}
</script>
</body> </html>
```

If … else statements – good morning example

Exercise

Write a script to check someone's age and determine if the person can consume alcohol

to the minimum legal drinking age is 21

If … else statements – good morning example

Exercise

Write a script to print “DES 350 class day” if current day is M or W

If statement- example

```
var score = 75;
var msg;
if (score >= 50) {
msg = 'Congratulations!';
msg += ' Proceed to the next round.';
}

var el = document.getElementById('answer')
el.textContent = msg;
```

If statement- example

```
<!DOCTYPE html>
<html>
 <head>
 <title>- If Statement</title>
<link rel="stylesheet" href="css/c04.css" />
</head> <body>
 <section id="page1">
 <h1>Bullseye</h1>
<img src="images/teacher.png" id="teacher" alt="teacher" /> <section id="answer">
</section>
</section>
<script src="js/if-statement.js"></script>
</body> </html>
```

If statement- example

BULLSEYE!

TARGET PRACTICE FOR YOUR MIND

Congratulations!
Proceed to the next
round.

If …else statement- example

```
var pass = 50;
 var score = 75;
var msg;
if (score > pass) {
msg = 'Congratulations, you passed!';
 } else {
msg = 'Have another go!';
 }
var el = document.getElementById('answer');
el.textContent = msg;
```

If …else statement with function- example

```
var score = 75;
var msg = '';
function congratulate() {
        msg += 'Congratulations! ';
 }
 if (score >= 50) {
        congratulate();
        msg += 'Proceed to the next round.';
 }
var el = document.getElementById('answer');
 el.innerHTML = msg;
```

JavaScript Events

- Interactions create events
- Events trigger code
- Code responds to users

- Events are used in combination with functions
- Events trigger functions to be executed
- (such as when a user clicks a button)

UI Events

onerror	The event occurs when an error occurs while loading an external file
onload	The event occurs when an object has loaded
onresize	The event occurs when a document view is resized
onscroll	The event occurs when an element's scrollbar is being scrolled
onunload	The event occurs once a page has unloaded (for <body

UI Events

```
<html>
<head>
<title>Random Script</title>
<script>
var myPix = new
Array("images/red.gif","images/green.gif","images/blue.gif")
function choosePic() {
      if(document.images{
        randomNum = Math.floor(Math.random()*myPix.length)
        document.image.src=myPix[randomNum]
           }}
</script>
```

UI Events

```
</script>
</head>
<body onLoad="choosePic()">

<img src="images/red.gif" name ="image">
</body>
</html>
```

Keyboard Events

onkeydown	The event occurs when the user is pressing a key
onkeypress	The event occurs when the user presses a key
onkeyup	The event occurs when the user releases a key

```
<input type="text" onkeypress="myFunction()">
```

Mouse Events

onclick	The event occurs when the user clicks on an element
ondblclick	The event occurs when the user double-clicks on an element
onmousedown	The event occurs when a user presses a mouse button over an element
onmouseover	The event occurs when the pointer is moved onto an element, or onto one of its children
onmouseout	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children
onmouseup	The event occurs when a user releases a mouse button over an element

Mouse Events

```
<html> <head>
<title>bgcolor_change</title>
<script language="JavaScript">
function newbg(thecolor) {
document.bgColor=thecolor;
}
</script> </head>
<body textcolor="black" link="black" alink="black”>
<a href="#" onmousedown="newbg('olive');"> olive</a><br />
<a href="#" onmousedown="newbg('blue');"> blue</a><br />
<a href="#" onmousedown="newbg('Beige');"> beige</a><br />
</body> </html>
```

Focus Events

onblur	The event occurs when an element loses focus
onfocus	The event occurs when an element gets focus

Focus Events

```
<!DOCTYPE html>
<html>
<body>
Enter your name: <input type="text" onfocus="myFunction(this)”>
<p>When the input field gets focus, a function is triggered which chan
the background-color.</p>
<script>
function myFunction(x) {
    x.style.background = "yellow";
}
</script>
</body></html>
```

Form Events

onchange	The event occurs when the content of a form element, the selection, or the checked state have changed (for <input>, <keygen>, <select>, and <textarea>)
oninput	The event occurs when an element gets user input
onreset	The event occurs when a form is reset
onsearch	The event occurs when a user writes something in a search field (for <input="search">)
onselect	The event occurs after the user selects some text (for <input> and <textarea>)
onsubmit	The event occurs when a form is submitted

Form Events

```
<!DOCTYPE html>
<html> <body>
<p>Write something in the text field to trigger a function.</p>
<input type="text" id="myInput" oninput="myFunction()”>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.getElementById("myInput").value;
    document.getElementById("demo").innerHTML = "You wrote: " + x;
}
</script> </body>
```

How Events trigger JavaScript code

1. Select element (<body>)
2. Specify event (onLoad)
3. Call code (function ())

DOM event handlers

element. on*event* = functionName;

```
function checkUsername() {
 //some code to check the length of username
}

var el = document.getElementById('username');
el.onblur =checkUsername;
```

DOM event handlers

Example
event_handler.html

DOM event handlers

Assignment 6

Write a script to print “DES 350 class” if current day is M or W.
Create a function to determine current day using Date object.

Use DOM event handler to print the message.

Switch statement

Switch statement starts with a variable switch value.
Each case indicates a possible value for the switch variable and the code that should run if the variable matches that value.

If a match is found, that code is executed. The break statement stops switch statement.

Better performance than multiple if statements.

Switch statement

```
switch (level) {                    //switch value variable
        case 'One':             //if switch value is "One' this code executed

        title='Level 1';
        break;

        case 'Two':             //if switch value is 'Two' this code executed
        title='Level 2';
        break;

        default:                //if none of the above this code executed
```

Switch statement- example

```
var msg;
var level = 2;
switch (level) {
case 1:
msg = 'Good luck on the first test';
break;
case 2:
msg = 'Second of three - keep going!';
break;
case 3:
msg = 'Final round, almost there!';
break;
default:
msg = 'Good luck!'; break;
 }
```

Switch statement

```
var el = document.getElementById('answer');
el.textContent = msg;
```

Switch statement

```
<!DOCTYPE html>
<html>
<head>
<title>Switch Statement</title>
 <link rel="stylesheet" href="css/c04.css" />
 </head>
<body>
 <section id="page1">
 <h1>Bullseye</h1>
<img src="images/teacher.png" id="teacher" alt="teacher" />
 <section id="answer"></section>
</section>
<script src="js/switch-statement.js"></script>
</body>
 </html>
```

Weak typing

JavaScript allows you not to specify what data type each variable will
(in declaration). JavaScript uses weak typing.

Data type for a value can change.

Data type	Purpose
string	Text
number	Number
boolean	true or false
null	Empty value
undefined	variable has been declared but not yet assigned a value

Type Coercion

Converts data types behind the scenes to compete the operation.

('1' >0) returns true

String is converted to a number

('ten'/2) returns NaN (Not a Number)

Type Coercion

Because of type coercion, the strict equality operators === and !==
Result in fewer unexpected values than == and != do.
false, 0 and ‘ ‘

(0 == ‘ ‘) true
(0 === ‘ ‘) false

(false == ‘ ‘) true
(false === ‘ ‘) false

For Loop

Loop checks a condition. If the condition is true, the statements in curly braces will be executed. The cycle repeats until the condition returns false.

```
for (var i = 0; i < 10; i++)  {
        document.write(i);
}
```

(initialization; condition; increment)

For Loop

Often used to loop through the items in an array.

<html>
<head>
<title>loop</title>
<script>

For Loop

```
function myFunction() {
var x="";
for (i=0;i<50;i++) {
  x=x + "The number is " + i + "<br>";
  }
document.getElementById("demo").innerHTML=x;
}
</script>
</head>
<body>
```

For Loop

```
<p>Click the button to loop through a block of as long as <em>i</em>
less than 50.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>

</body>
</html>
```

while Loop

While loop will run as long as the condition is true.

```
while ( i < 10 )  {
        statements;
        i ++;
}
```

do while Loop

Do while loop will execute statements first, before it checks the condition .

```
do {
        statements;
        i ++;
} while ( i < 10 );
```

do while Loop - example

```
var i = 1; // Set counter to 1
var msg = ''; // Message // Store 5 times table in a variable
do {
msg += i + ' x 5 = ' + (i * 5) + '<br />';
i++;
}  while (i < 1); // Note how this is already 1 and it still runs
document.getElementById('answer').innerHTML = msg;
```

do while Loop - example

```
<!DOCTYPE html>
<html>
  <head>
    <title>Do While Loop</title>
    <link rel="stylesheet" href="css/c04.css" />
  </head>
  <body>
          <section id="page1">
                  <h1>Bullseye</h1>
                  <img src="images/teacher.png" id="teacher" alt="teacher" />
                  <section id="answer"></section>
            </section>
    <script src="js/do-while-loop.js"></script>
  </body> </html>
```

3 ways to use events

1) HTML event handlers – old fashioned

```
<a onclick="hide()">
```

2) DOM event handlers

```
var el = document.getElementById('username');
el.onblur = checkUsername();
```

3) Event listeners

```
var el = document.getElementById('username');
el.addEventListener('blur', checkUsername, false);
```

Event Listeners

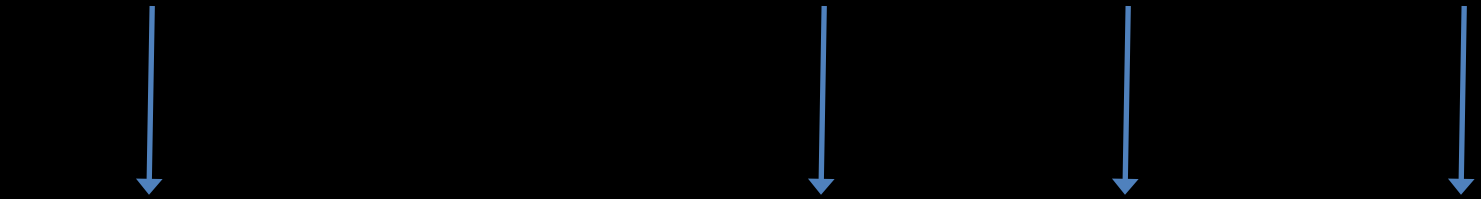
Most recent approach

Can call more than one function at a time

Not supported by older browsers

Event Listeners

element.addEventListener (‘event’, functionName, [Boolean]);



DOM element ‘blur’ checkUserName false

Indicates capture

Usually set to false

Event Listeners – event-listener.html

```
function checkUsername() {
var elMsg = document.getElementById('feedback');
if (this.value.length < 5) {
elMsg.textContent = 'Username must be 5 characters or more';
} else {
elMsg.textContent = '';
} }
var elUsername = document.getElementById('username');
elUsername.addEventListener('blur', checkUsername, false);
```

Event Listeners – event-listener-with-ie-fallback.html

IE 5-8 did not support event listeners

Example fallback

attachEvent() method

Event Listeners – event-listener-with-ie-fallback.html

```
if (elUsername.addEventListener) {
  elUsername.addEventListener('blur', function(){    checkUsername(5);
}, false);
} else {
elUsername.attachEvent('onblur', function(){
checkUsername(5);
});
}
```

Types of Events

W3C DOM Events

HTML5 Events

Under development

submit

Input

change

hashchnage

BOM Events

Touchscreen devices, accelerometer, etc.

touchstart

touched

orientationchange

UI Events

error	The event occurs when an error occurs while loading an external file
load	The event occurs when an object has loaded
resize	The event occurs when a document view is resized
scroll	The event occurs when an element's scrollbar is being scrolled
unload	The event occurs once a page has unloaded (for <body>)

UI Events

Example- load.html

```
function setup() {
var textInput;
textInput = document.getElementById('username');
input
  textInput.focus();
}
window.addEventListener('load', setup, false);
```

Focus & Blur Events

blur	The event occurs when an element loses focus
focus	The event occurs when an element gets focus
focusin	same as focus
focusout	same as blur

Focus & Blur Events

Example: focus-blur.html

A the text input gains and loses focus,
the feedback is shown to the user in the element below

Mouse Events

click	The event occurs when the user clicks on an element
dblclick	The event occurs when the user double-clicks on an element
mousedown	The event occurs when a user presses a mouse button over an element
mouseover	The event occurs when the pointer is moved onto an element, or onto one of its children
mouseout	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children
mouseup	The event occurs when a user releases a mouse button over an element
mousemove	occurs when the cursor is moved around the element

Mouse Events

Example: click.html

Where Events Occur

screen	page	client
screenX	pageX	clientX
screenY	pageY	clientY

Where Events Occur

Example: position.html

Where Events Occur

```
var sx = document.getElementById('sx');
var sy = document.getElementById('sy');
var px = document.getElementById('px')
 var py = document.getElementById('py')
 var cx = document.getElementById('cx')
 var cy = document.getElementById('cy')

function showPosition(event) {
sx.value = event.screenX;
sy.value = event.screenY;
px.value = event.pageX;
py.value = event.pageY;
cx.value = event.clientX;
cy.value = event.clientY;
}
var el = document.getElementById('body');
el.addEventListener('mousemove', showPosition, false);
```

Keyboard Events

input	Occurs when the value of <input> or <textarea> chnages
keydown	The event occurs when the user is pressing a key
keypress	The event occurs when the user presses a key
keyup	The event occurs when the user releases a key

Keyboard Events

Example: keypress.html

Event listener checks for keypress event on the <textarea> element

Each time it fires, the charCount() function updates the character count and shows the last character used.

Keyboard Events

```
var el;
function charCount(e) {
var textEntered, charDisplay, counter, lastkey;
textEntered = document.getElementById('message').value;
charDisplay = document.getElementById('charactersLeft');
counter = (180 - (textEntered.length));
charDisplay.textContent = counter;
lastkey = document.getElementById('lastKey');
lastkey.textContent = 'Last key in ASCII code: ' + e.keyCode;
}
el = document.getElementById('message');
el.addEventListener('keypress', charCount, false);
```

Form Events

submit	Occurs when form is submitted
change	occurs when the status of forms change (ex. radio button selected)
input	occurs when user types in text in the <input> or <textarea>

Form Events

Example: form.html

The change events triggers the PackageHint() function.

JavaScript Events

User Interface events (load)

Focus & Blur

Mouse Events

Keyboard events

Form events

Mutation events

HTML5 events

Mouse Events

click	The event occurs when the user clicks on an element
dblclick	The event occurs when the user double-clicks on an element
mousedown	The event occurs when a user presses a mouse button over an element
mouseover	The event occurs when the pointer is moved onto an element, or onto one of its children
mouseout	The event occurs when a user moves the mouse pointer out of an element, or out of one of its children
mouseup	The event occurs when a user releases a mouse button over an element
mousemove	occurs when the cursor is moved around the element

Mouse Events

Example: click.html

Keyboard Events

input	Occurs when the value of <input> or <textarea> chnages
keydown	The event occurs when the user is pressing a key
keypress	The event occurs when the user presses a key
keyup	The event occurs when the user releases a key

Keyboard Events

Example: keypress.html

Event listener checks for keypress event on the <textarea> element

Textfiled is limited to 180 characters

Each time it fires, the charCount() function updates the character count and shows the last character used.

Each key is shown in ASCII code

HTML ASCII Code

ASCII = American Standard Code for Information Interchange

It was designed in the early 60's, as a standard character set for computers and electronic devices.

ASCII is a 7-bit character set containing 128 characters.

It contains the numbers from 0-9, the upper and lower case English letters from A to Z, and some special characters.

The character sets used in modern computers, in HTML, and on the Internet, are all based on ASCII.

HTML ASCII Code

Char	Number	Description
A	65	uppercase A
B	66	uppercase B
C	67	uppercase C
D	68	uppercase D
E	69	uppercase E
F	70	uppercase F
G	71	uppercase G
H	72	uppercase H

http://www.w3schools.com/charsets/ref_html_ascii.asp

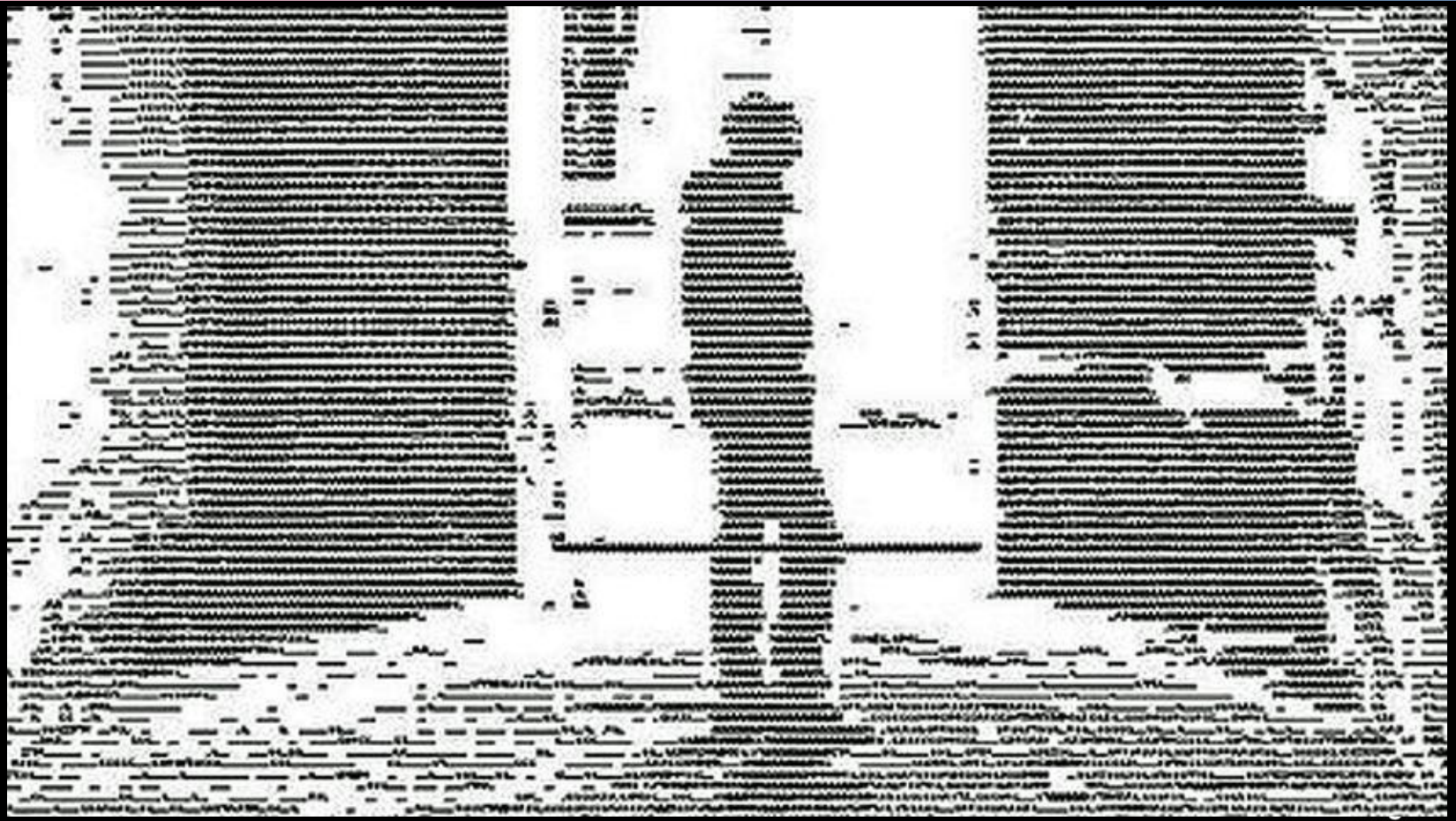
ASCII Art

ASCII art is an early graphic-design technique, dating back to the 1890s when typewriters became more than just a new tool for writing.

School of Design

ASCII Art

ASCII Art



Keyboard Events

```
var el;
function charCount(e) {
var textEntered, charDisplay, counter, lastkey;
textEntered = document.getElementById('message').value;
charDisplay = document.getElementById('charactersLeft');
counter = (180 - (textEntered.length));
charDisplay.textContent = counter;

lastkey = document.getElementById('lastKey');
lastkey.textContent = 'Last key in ASCII code: ' + e.keyCode;  }

el = document.getElementById('message');
el.addEventListener('keypress', charCount, false);
```

Form Events

submit	Occurs when form is submitted
change	occurs when the status of forms change (ex. radio button selected)
input	occurs when user types in text in the <input> or <textarea>

Form Events

Example: form.html

The change events triggers the PackageHint() function.
This shows different messages below the menu to reflect the choice.

Form Events

Exercise: modify the form.html and javascript to offer the user a drop down menu with two choices : 20 or younger / 21 or older.

When the use interacts with the drop-down menu, the messages below say “ You can drink” or “You cannot drink”

Mutation Events

When elements are added to or removed from the DOM, its structure changes.

These changes trigger mutation events.

Mutation observers are designed to wait until a script has finished its task before reacting, then report the changes as a batch (rather than one at a time).

Mutation Events

DOMNodeInserted	Fires when a node is inserted into the DOM tree e.g. using appendChild(), replaceChild(), or insertBefore()
DOMNodeRemoved	Fires when a node is removed from the DOM tree e.g. removeChild()
DOMSubtreeModified	Fires when the DOM structure changes after the two events listed above occur.

Mutation Events

DOMNodeInsertedIntoDocument	Fires when a node is inserted into the DOM tree as a descendant of another node that is already in the document.
DOMNodeRemovedFromDocument	Fires when a node is removed into the DOM tree as a descendant of another node that is already in the document.

Mutation Events

Example: mutation.html

Two event listeners each trigger their own function.

The first listens for when the user clicks the link to add a new list item.

It then uses DOM manipulation events to add a new element (changing the DOM structure and triggering mutation events).

The second event listener waits for the DOM tree within the <ul> element to change. When the DOMNodeInserted event fires, it calls a function updateCount(). This function counts how many items there are in the list, and then updates the list count at the top of the page accordingly.

HTML5 Events

DOMContentLoaded	Fires when the DOM tree is formed (images, css and js might be still loading). Script starts to run earlier than using load event which waits for other resources such as images to load.
hashchnage	Fires when the URL hash changes (w/o the entire window refreshing). Hashes are used on links to specific parts (known as anchors) within a page and also on pages that use AJAX to load content.

HTML5 Events

Anchor link

```
<a id="tips">Useful Tips Section</a>

<a href="#tips">Visit the Useful Tips Section</a>
```

HTML5 Events

beforeunload	Fires on the window object before the page is unloaded. It should only be used to help the user (not to encourage them to stay on the website if they are trying to leave). Example. It could be helpful to let a user know that changes on a form he competed have not been saved.

HTML5 Events

Example: html5_events.html

As soon as the DOM tree has been formed the focus is given to the text input.

The DOMContentLoaded event fires before the load event.

If users try to leave the page before they press submit button, the beforeunload event checks that they want to leave.

HTML5 Events

Example.html

Shows an interface for a user to record voice notes.

The user can enter a name which is displayed in the heading,

and they can press record (which changes the image that is shown).

Keyup event triggers a function writeLabel() which copies the text from the form input and writes it into the main heading.

The record button has an attribute data-state. When the page loads, its value is record. When user presses the button, the value of this attribute changes to pause (and triggers a new css rule to indicate that it is now recording).

JavaScript Events

Events are the browser's way of indicating when something has happened (e.g. a page has finished loading, button has been clicked)

When an event occurs on an element, it can trigger a JavaScript function. When this function then changes the web page in some way, it feels interactive because it has responded to the user.

You can use event delegation to monitor for events that happen on all of the children of an element.

The most commonly used events are W3C DOM events, although there are others in theHTML5 specifications as well as browser-specific events.

JavaScript Events

Assignment: JavaScript events.

Write a script that will calculate a 15% and a 20% tip.

The users type into a text field the amount that they wish to tip on.

The user selects 15% or 20% they wish to tip via a drop-down menu.

When they click on a "Calculate" button the amount of the tip and the total amount (tip plus original amount) are displayed.