Client-Side Web Development

Class 9.1

Today's Topics

- For... of Loop

- Retrieving Multiple Elements

- Conditional Statements

- **Exercise:** A Horse of a Different Color

Announcements

Schedule Change

Web Portfolio

Any Questions?

The `for... of` **Loop**

For... of loop **is used to iterate over arrays**

```javascript
// Create an array of animals
const $animals = ['Cow', 'Pig', 'Duck']

// Iterate over the array using for... of
for (const $animal of $animals) {
  console.log(`Old McDonald has a ${$animal}.`)
}
```

```javascript
// Create an array of animals
const $animals = ['Cow', 'Pig', 'Duck']

// Iterate over the array using for... of
for (const $animal of $animals) {
  console.log(`Old McDonald has a ${$animal}.`)
}
```

Variable

It is common to make the variable name be the singular version of the array variable name

```javascript
// Create an array of animals
const $animals = ['Cow', 'Pig', 'Duck']

// Iterate over the array using for... of
for (const animal of $animals) {
  console.log(`Old McDonald has a ${$animal}.`)
}
```

The array

```javascript
// Create an array of animals
const $animals = ['Cow', 'Pig', 'Duck']

// Iterate over the array using for... of
for (const $animal of $animals) {
  console.log(`Old McDonald has a ${$animal}.`)
}
```

The variable is assigned
a different value for each iteration

Value of 'animal' for
each iteration
1. Cow
2. Pig
3. Duck

Retrieving Multiple Elements

The `querySelectorAll()` Method

This method will return an array of all the elements that match the provided CSS selector

If no element is found, an empty array will be returned.

```html
<div id="boxes">
  <div class="box"></div>
  <div class="box"></div>
  <div class="box"></div>
  <div class="box"></div>
</div>
```

```javascript
const $boxes = document.querySelectorAll('.box')

// Add class to the third box
$boxes[2].classList.add('red')
```

```
const $boxes = document.querySelectorAll('.box')



// Add class to the third box
$boxes[2].classList.add('red')
```

Is an array

The querySelectorAll() method returns an array of HTML elements

```
const $boxes = document.querySelectorAll('.box')

// Add class to the third box
$boxes[2].classList.add('red')
```

The third item in the array

Each HTML element can be
retrieved from the array using
the appropriate index

```javascript
const $boxes = document.querySelectorAll('.box')

// Add class to all boxes
for (const $box of $boxes) {
  $box.classList.add('red')
}
```

Conditional Statements

Conditional statements determine when and where to execute code based on conditions

If Statement

The *if statement* is the most basic conditional statement

The *if statement* has an expression and a block of code

If the expression result is `true`, the code block will execute

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
}


if (sound === 'Moo') {
  console.log(`It was a cow.`)
}
```

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
}
```

This expression is false

```javascript
if (sound === 'Moo') {
  console.log(`It was a cow.`)
}
```

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
}
```

This line will NOT execute

```javascript
if (sound === 'Moo') {
  console.log(`It was a cow.`)
}
```

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
}


if (sound === 'Moo') {
  console.log(`It was a cow.`)
}
```

This expression is true

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
}


if (sound === 'Moo') {
  console.log(`It was a cow.`)
}
```

This line will execute

If / Else Statement

The **else** is used to execute an alternate code block in the event the if expression evaluate to `false`

```javascript
const chicken = false

// Is it a chicken?
if (chicken) {
  console.log(`It is a chicken.`)
} else {
  console.log(`It is not a chicken`)
}
```

```javascript
const chicken = false

// Is it a chicken?
if (chicken) {
  console.log(`It is a chicken.`)
} else {
  console.log(`It is not a chicken`)
}
```

```javascript
const chicken = false


// Is it a chicken?
if (chicken) {

  console.log(`It is a chicken.`)
} else {

  console.log(`It is not a chicken`)

}
```

Go to the else statement

```javascript
const chicken = false


// Is it a chicken?
if (chicken) {
  console.log(`It is a chicken.`)
} else {
  console.log(`It is not a chicken`)
}
```

This line will execute

If / Else If Statement

The *else if* shorthand can be used to combine multiple if / else statements together

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
} else if (sound === 'Moo') {
  console.log(`It was a cow.`)
} else if (sound === 'Cluck') {
  console.log(`It was a chicken.`)
}
```

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
} else if (sound === 'Moo') {
  console.log(`It was a cow.`)
} else if (sound === 'Cluck') {
  console.log(`It was a chicken.`)
}
```

If false, then continue
to the next expression

```javascript
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
} else if (sound === 'Moo') {
  console.log(`It was a cow.`)
} else if (sound === 'Cluck') {
  console.log(`It was a chicken.`)
}
```

If true, then ignore all future expression

```
const sound = 'Moo'

// Which animal made that sound
if (sound === 'Oink') {
  console.log(`It was a pig.`)
} else if (sound === 'Moo') {
  console.log(`It was a cow.`)
} else if (sound === 'Cluck') {
  console.log(`It was a chicken.`)
}
```

This statement is never evaluated

Exercise: A Horse of a Different Color

For next class...

- Work period

For Next Week...

- Functions

- Event Handling