



# Video 2.1

## Chris Murphy

# Motivation

---

- HTML and CSS only allow for **static** content
- HTML/CSS do not allow for generating **dynamic** content that can change based on user input, activity, etc.
- However, recall that the browser has an engine for generating dynamic content using **JavaScript**

# JavaScript

---

- Developed at Netscape Communications in mid-1990s as a way of adding dynamic elements to HTML
- Originally known as “LiveScript”; changed to “JavaScript” soon after its release
- Now one of the most popular programming languages in the world

# Developing in JavaScript

---

1. JavaScript can be embedded directly in the HTML inside `<script>` tags and/or using `<link>` tags to external `.js` files
2. Browsers such as Chrome provide a JavaScript “REPL” console for writing and evaluating code
  - Can also see output generated by JavaScript in HTML
3. You can also develop JavaScript in a `.js` file and execute it in a runtime environment such as Node.js

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    This is my first JavaScript web page.
    <p>

      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

    </body>
  </html>
```

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    This is my first JavaScript web page.
    <p>

      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

  </body>

</html>
```

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    This is my first JavaScript web page.
    <p>

      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

    </body>
  </html>
```

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    This is my first JavaScript web page.
    <p>

      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

    </body>
  </html>
```



```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <body>
    This is my first JavaScript web page.
    <p>

      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

    </body>
  </html>
```

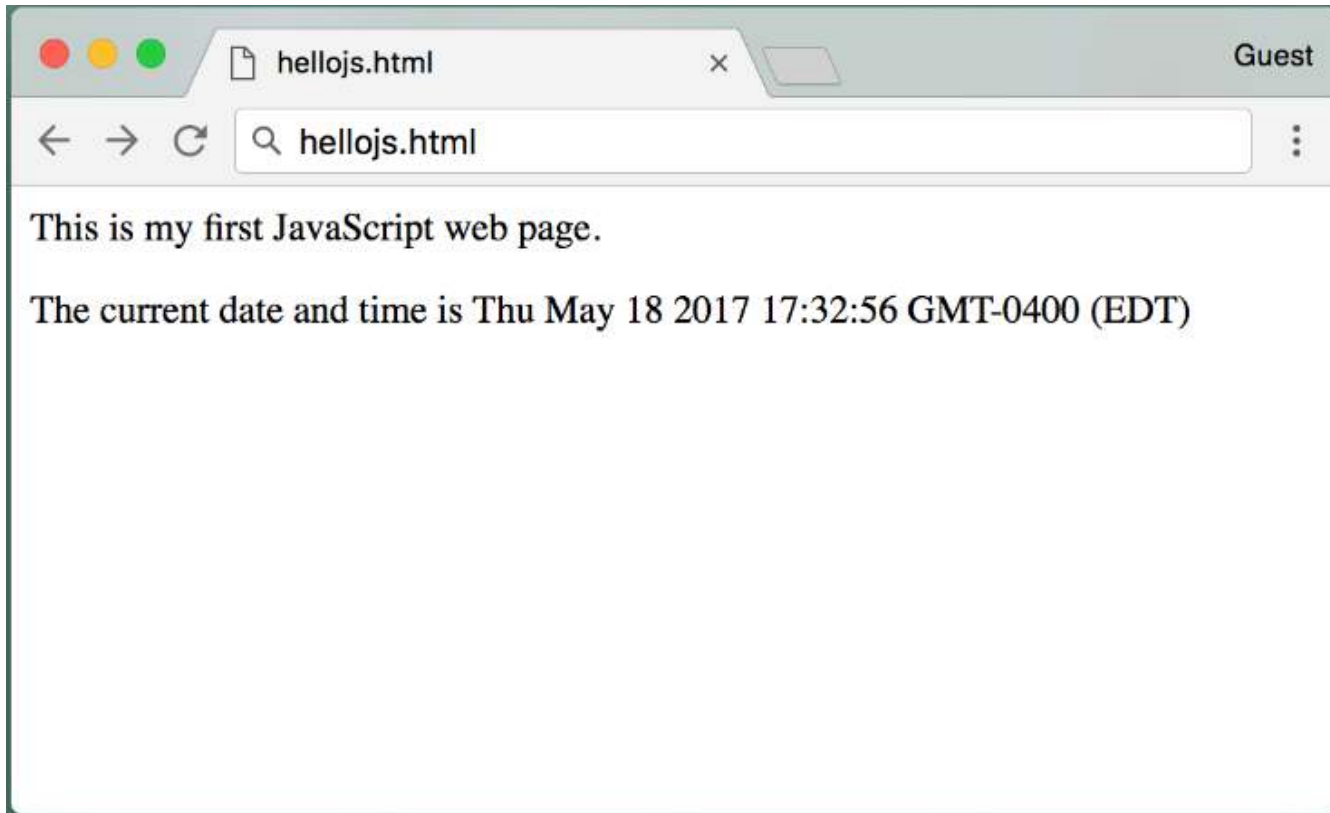
```
<!DOCTYPE html>
<html>
  <head>
  </head>

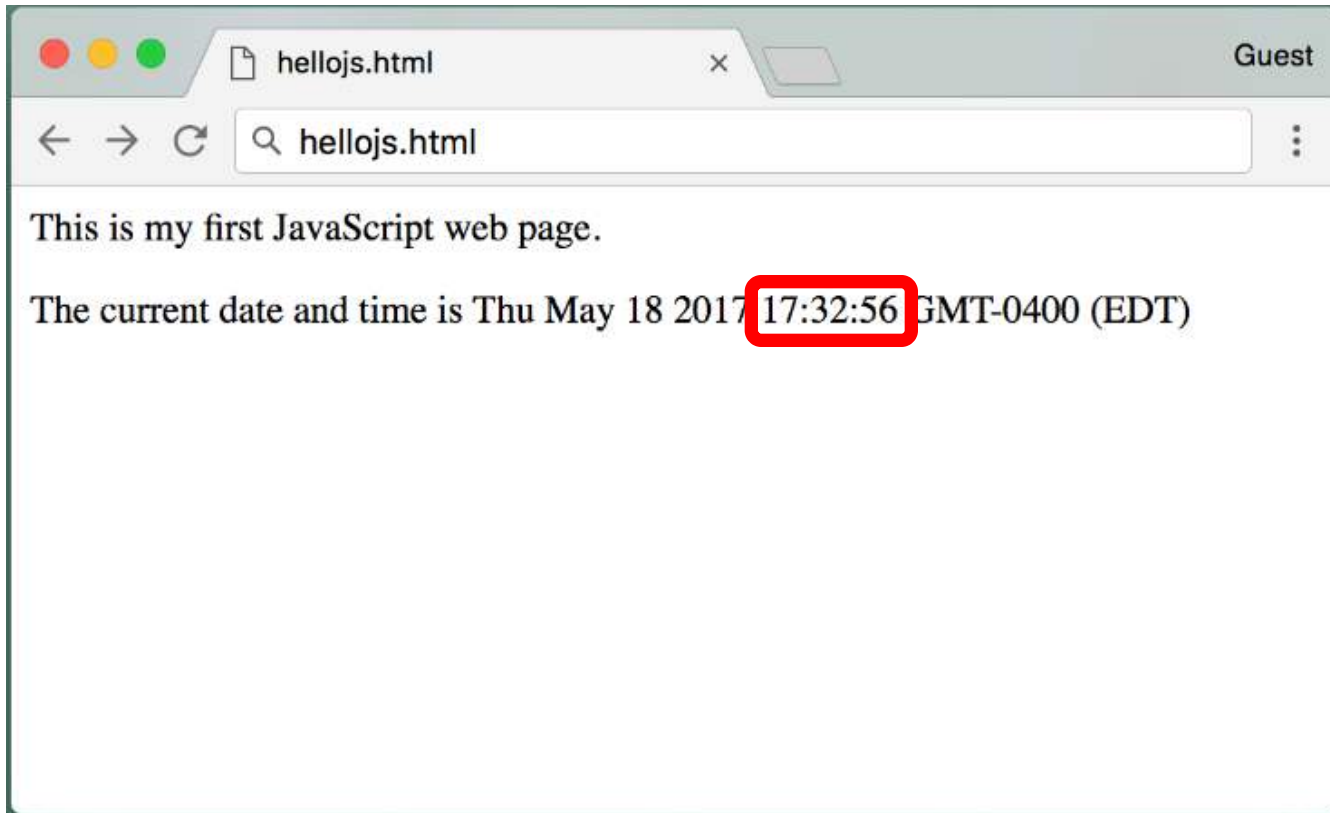
  <body>
    This is my first JavaScript web page.
    <p>

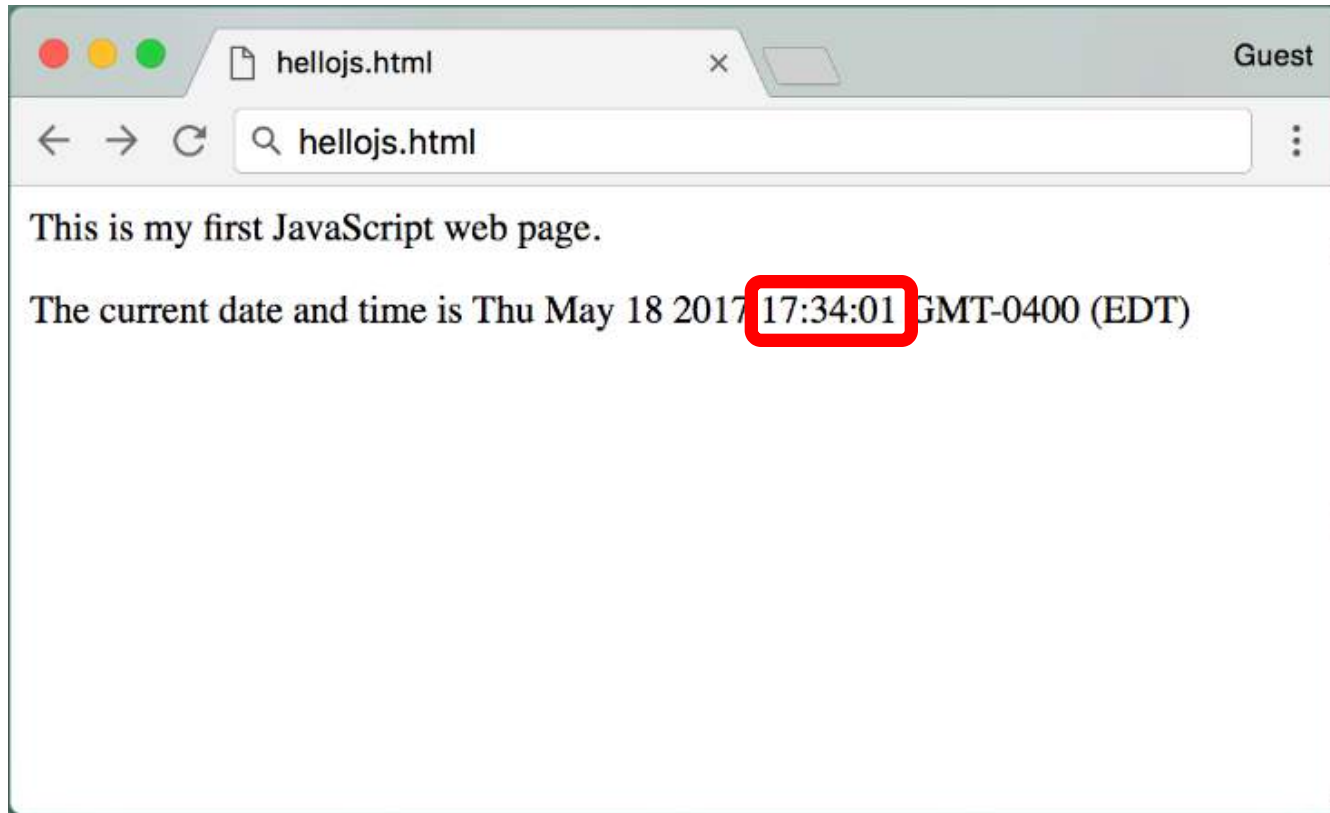
      <script>
        document.write('The current date and time is ');
        var time = new Date();
        document.write(time);
      </script>

    </body>

  </html>
```







# Looking Ahead

---

- JavaScript basics
- How JavaScript code interacts with HTML elements
- Frameworks for developing “client-side” JavaScript (i.e., in the browser)
- Frameworks for developing “server-side” JavaScript (i.e., on a server)

# Disclaimer!

---

- The code and examples for this part of the course have been developed using:
  - JavaScript 1.7
  - Google Chrome 58.0
  - Mac OSX 10.11.5
- You may observe slight differences on your own platform



# Video 2.2

## Chris Murphy



# JavaScript Basics

---

- Like many other programming languages, JavaScript includes:
  - variables, arrays, and objects
  - loops and conditional statements
  - functions
- Even if you know Java, there are still some important differences
  - defining functions and objects
  - interacting with HTML

# Declaring a Variable

---

- The basic syntax for declaring any JavaScript variable is `var variableName = ...`

```
var age = 22;  
  
var name = 'Jane Doe';  
  
var isMale = false;
```

# Declaring a Variable

---

- The basic syntax for declaring any JavaScript variable is `var variableName = ...`

```
var age = 22;  
  
var name = 'Jane Doe';  
  
var isMale = false;
```

# Declaring a Variable

---

- The basic syntax for declaring any JavaScript variable is `var variableName = ...`

```
var age = 22;  
var name = 'Jane Doe' ;  
var isMale = false;
```

# Declaring a Variable

---

- The basic syntax for declaring any JavaScript variable is `var variableName = ...`

```
var age = 22;  
var name = 'Jane Doe';  
var isMale = false;
```

# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write(var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write(age);  
</script>
```

# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write (var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write (age);  
</script>
```

# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write (var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write (age);  
</script>
```



# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write (var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write (age) ;  
</script>
```

# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write (var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write (age);  
</script>
```

My age is: 12

# Viewing a variable's value (1)

---

- If using a `<script>` section in a HTML file, or an external .js file, `document.write (var)` will display a variable's value in the HTML

```
My age is:  
<script>  
  var age = 12;  
  document.write (age);  
</script>
```

My age is: 12

- However, this approach is discouraged
- We will see better alternatives later!

# Viewing a variable's value (2)

---

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>
  var age = 12;
  console.log(age);
</script>
```

# Viewing a variable's value (2)

---

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>  
  var age = 12;  
  console.log(age);  
</script>
```

# Viewing a variable's value (2)

---

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>  
  var age = 12;  
  console.log(age);  
</script>
```

# Viewing a variable's value (2)

---

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>
  var age = 12;
  console.log(age) ;
</script>
```

# Viewing a variable's value (2)

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>
  var age = 12;
  console.log(age);
</script>
```





# Viewing a variable's value (2)

- You can also use `console.log(var)` to print a variable's value in the browser's JavaScript console

```
<script>
  var age = 12;
  console.log(age);
</script>
```



# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>
  var age = 12;
  alert (age);
</script>
```

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>  
  var age = 12;  
  alert (age);  
</script>
```

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>  
  var age = 12;  
  alert (age);  
</script>
```

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>
  var age = 12;
  alert (age) ;
</script>
```

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>  
  var age = 12;  
  alert (age);  
</script>
```

This page says:

12

OK

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>
  var age = 12;
  alert (age);
</script>
```

This page says:

12

OK

- Last, if using the browser JavaScript console (REPL), just type the name of the variable

```
> var age = 12;
```

# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>
  var age = 12;
  alert (age);
</script>
```

This page says:

12

OK

- Last, if using the browser JavaScript console (REPL), just type the name of the variable

```
> var age = 12;
> age
```



# Viewing a variable's value (3)

---

- Also, `alert (var)` will create a popup with the variable's value that appears on top of the browser

```
<script>
  var age = 12;
  alert (age);
</script>
```

This page says:

12

OK

- Last, if using the browser JavaScript console (REPL), just type the name of the variable

```
> var age = 12
> age
12
```

# Changing a variable's type

---

- The type of each variable does not need to be specified and can be changed at any time.

```
var id = 33.2;  
  
id = 'secret';
```

# Changing a variable's type

---

- The type of each variable does not need to be specified and can be changed at any time.

```
var id = 33.2;  
  
id = 'secret';
```

# Changing a variable's type

---

- The type of each variable does not need to be specified and can be changed at any time.

```
var id = 33.2;  
  
id = 'secret';
```

# Primitive Types

---

Type	Example values
Number	<code>5, 1.25, 1.1e5, +Infinity, -Infinity, NaN</code>

# Primitive Types

---

Type	Example values
Number	<code>5, 1.25, 1.1e5, +Infinity, -Infinity, NaN</code>
String	<code>'hello'</code>

# Primitive Types

---

Type	Example values
Number	<code>5, 1.25, 1.1e5, +Infinity, -Infinity, NaN</code>
String	<code>'hello'</code>
Boolean	<code>true, false</code>

# Primitive Types

---

Type	Example values
Number	5, 1.25, 1.1e5, +Infinity, -Infinity, NaN
String	'hello'
Boolean	true, false
Null	null



# Primitive Types

---

Type	Example values
Number	<code>5, 1.25, 1.1e5, +Infinity, -Infinity, NaN</code>
String	<code>'hello'</code>
Boolean	<code>true, false</code>
Null	<code>null</code>
Undefined	<code>undefined</code>

# Numbers

---

- All JavaScript numbers are stored using floating-point notation
  - i.e. 5 is stored internally as 0.5e1
- `+infinity` represents all numbers greater than `Number.MAX_VALUE` (around  $10^{308}$ )
- `-infinity` represents all numbers less than `Number.MIN_VALUE` (around  $10^{-324}$ )
- NaN represents any non-number value
  - `Number('tree')` would return NaN

# Number Operations

---

- Basic arithmetic (+, -, \*, /, %) can be used on JavaScript numbers
- Precedence will follow MDAS unless parentheses are used
- ++ and -- can be used to increment/decrement JavaScript numbers

```
var a = 4;
a++;           // a = 5
var c = a - 3; // 2
var d = c + 3 * a; // 17
var e = ( c + 3 ) * a; // 25
```

# Strings

---

- JavaScript strings are series of 16-bit unsigned integers, each integer representing a character
- Convention is to use single quotes for strings unless single quotes exist within the string
  - `'I am a dolphin'` vs. `"I'm a dolphin"`
- Escape characters use backslash: `'\n \t \\'`
- All JavaScript strings are immutable
  - Any manipulation results in a new string

# String Functions

---

- `+` or `.concat(otherString)` can be used to concatenate strings (add them together)

```
var firstName = 'John';  
var lastName = 'doe';
```

# String Functions

---

- `+` or `.concat(otherString)` can be used to concatenate strings (add them together)

```
var firstName = 'John';  
var lastName = 'doe';  
  
var fullName= firstName.concat(' ', lastName); // 'John doe'
```

# String Functions

---

- `+` or `.concat(otherString)` can be used to concatenate strings (add them together)

```
var firstName = 'John';  
var lastName = 'doe';  
  
var fullName= firstName.concat(' ', lastName); // 'John doe'  
var greeting = 'HELLO, ' + fullName;
```

# String Functions

---

- `+` or `.concat(otherString)` can be used to concatenate strings (add them together)
- `.toUpperCase()` and `.toLowerCase()` change the case of every character in a string

```
var firstName = 'John';  
var lastName = 'doe';  
  
var fullName= firstName.concat(' ', lastName); // 'John doe'  
var greeting = 'HELLO, ' + fullName;  
  
console.log(greeting.toUpperCase()); // 'HELLO, JOHN DOE'  
console.log(greeting.toLowerCase()); // 'hello, john doe'
```



# String Functions

---

- `+` or `.concat(otherString)` can be used to concatenate strings (add them together)
- `.toUpperCase()` and `.toLowerCase()` change the case of every character in a string
- `var.length` gets the length of a string

```
var firstName = 'John';  
var lastName = 'doe';  
  
var fullName= firstName.concat(' ', lastName); // 'John doe'  
var greeting = 'HELLO, ' + fullName;  
  
console.log(greeting.toUpperCase());           // 'HELLO, JOHN DOE'  
console.log(greeting.toLowerCase());           // 'hello, john doe'  
  
console.log(greeting.length);                 // 15
```

# Booleans

---

- Booleans are logical values that can only be `true` or `false`
- Any value can be used as a boolean in JavaScript
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...
- Any variable type can become a boolean when used with logical operators

# Null and Undefined

---

- **Null** is a value that can be assigned to variables to represent “no value”

```
var occupation = null;  
console.log(occupation); // null
```

# Null and Undefined

---

- **Null** is a value that can be assigned to variables to represent “no value”

```
var occupation = null;  
console.log(occupation); // null
```

- **Undefined** means that a variable was declared but no value has been assigned

```
var salary;  
console.log(salary); // undefined
```

# Summary

---

- JavaScript variables do not need to have their types specified when they are declared
- Variable types are allowed to change
- Five primitive types: number, string, boolean, null, undefined



## Video 2.3

### Chris Murphy

# Variables in JavaScript

---

- Five primitive types: number, string, boolean, null, undefined
- Sometimes we may want to have a collection of ordered values
- Sometimes we may want to have a collection of associated values with semantically meaningful names/keys

# Arrays

---

- Arrays are used to store a list of values in a single variable
- Values can be of any type, and are split with commas and wrapped in square brackets

```
var myArray = ['cars', 12, false];
```



# Arrays

---

- Arrays are used to store a list of values in a single variable
- Values can be of any type, and are split with commas and wrapped in square brackets
- Values can be accessed with `arrayVar[index]`

```
var myArray = ['cars', 12, false];  
  
var age = myArray[1];  
console.log(age);           // 12
```

# Arrays

---

- Arrays are used to store a list of values in a single variable
- Values can be of any type, and are split with commas and wrapped in square brackets
- Values can be accessed with `arrayVar[index]`

```
var myArray = ['cars', 12, false];  
  
var age = myArray[1];  
console.log(age);           // 12  
myArray[2] = true;  
console.log(myArray[2]);   // true
```

# Arrays

---

- Arrays are used to store a list of values in a single variable
- Values can be of any type, and are split with commas and wrapped in square brackets
- Values can be accessed with *arrayVar[index]*
- The length of an array can be found with *.length*

```
var myArray = ['cars', 12, false];  
  
var age = myArray[1];  
console.log(age);           // 12  
myArray[2] = true;  
console.log(myArray[2]);    // true  
  
console.log(myArray.length); //3
```

# Array Indices

---

- When **reading** an array value by its index, `arrayVar[index]` will return `undefined` if the index is out of bounds

```
var a = ['cat', 'dog', 'banana'];  
console.log(a[4]); // undefined  
console.log(a[-9]); // undefined
```

# Array Indices

---

- When **reading** an array value by its index, `arrayVar[index]` will return `undefined` if the index is out of bounds

```
var a = ['cat', 'dog', 'banana'];  
  
console.log(a[4]); // undefined  
  
console.log(a[-9]); // undefined
```

# Array Indices

---

- When **reading** an array value by its index, `arrayVar[index]` will return `undefined` if the index is out of bounds

```
var a = ['cat', 'dog', 'banana'];  
console.log(a[4]); // undefined  
console.log(a[-9]); // undefined
```

# Array Indices

---

- When **reading** an array value by its index, `arrayVar[index]` will return `undefined` if the index is out of bounds

```
var a = ['cat', 'dog', 'banana'];  
console.log(a[4]); // undefined  
console.log(a[-9]); // undefined
```

# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```



# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```

# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];  
  
a[4] = 'panda';  
console.log(a[4]); // "panda"  
console.log(a[3]); // undefined  
  
a[-5] = 'elephant';  
console.log(a[-5]); // "elephant"  
  
console.log(a);  
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```

# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```

# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```



# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```



# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```

# Array Indices

---

- When **writing** an array value by its index, `arrayVar[index]` will
  - add an element at that index if `index >= arrayVar.length`
  - create a mapping from the index to the element if `index < 0`

```
var a = ['cat', 'dog', 'banana'];

a[4] = 'panda';
console.log(a[4]); // "panda"
console.log(a[3]); // undefined

a[-5] = 'elephant';
console.log(a[-5]); // "elephant"

console.log(a);
// (5) ["cat", "dog", "banana", undefined × 1, "panda", -5: "elephant"]
```

# Adding to an Array

---

- Elements can be added to arrays using **push ()** and **unshift ()**
  - **push ()** will add elements to the end of the array
  - **unshift ()** will add elements to the beginning of the array

```
var myArray = ['car', 'bike'];  
  
myArray.push('scooter');  
console.log(myArray);           // car,bike,scooter  
  
myArray.unshift('train');  
console.log(myArray);           // train,car,bike,scooter
```



# Adding to an Array

---

- Elements can be added to arrays using **push ()** and **unshift ()**
  - **push ()** will add elements to the end of the array
  - **unshift ()** will add elements to the beginning of the array

```
var myArray = ['car', 'bike'];  
  
myArray.push('scooter');  
console.log(myArray);           // car,bike,scooter  
  
myArray.unshift('train');  
console.log(myArray);           // train,car,bike,scooter
```

# Adding to an Array

---

- Elements can be added to arrays using **push()** and **unshift()**
  - **push()** will add elements to the end of the array
  - **unshift()** will add elements to the beginning of the array

```
var myArray = ['car', 'bike'];  
  
myArray.push('scooter');  
console.log(myArray);           // car,bike,scooter  
  
myArray.unshift('train');  
console.log(myArray);           // train,car,bike,scooter
```

# Adding to an Array

---

- Elements can be added to arrays using `push()` and `unshift()`
  - `push()` will add elements to the end of the array
  - `unshift()` will add elements to the beginning of the array

```
var myArray = ['car', 'bike'];  
  
myArray.push('scooter');  
console.log(myArray);           // car,bike,scooter  
  
myArray.unshift('train');  
console.log(myArray);           // train,car,bike,scooter
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using **pop()** and **shift()**
  - **pop()** will remove and return an element from the end of the array
  - **shift()** will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];  
  
var vehicle = myArray.pop();  
console.log(vehicle);           // scooter  
console.log(myArray);          // train,car,bike  
  
vehicle = myArray.shift();  
console.log(vehicle);           // train  
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```



# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);          // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);           // train
console.log(myArray);          // car,bike
```

# Removing from an Array

---

- Elements can be removed from arrays using `pop()` and `shift()`
  - `pop()` will remove and return an element from the end of the array
  - `shift()` will remove and return an element from the beginning

```
var myArray = ['train', 'car', 'bike', 'scooter'];

var vehicle = myArray.pop();
console.log(vehicle);           // scooter
console.log(myArray);         // train,car,bike

vehicle = myArray.shift();
console.log(vehicle);         // train
console.log(myArray);         // car,bike
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id)  // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id)  // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id)  // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id) // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id)  // 2984
```



# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age);           // 25  
console.log(person['company'].id)  // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {
  name: 'John Doe',
  age: 25,
  isMale: true,
  personality: ['patient', 'loyal', 'happy'],
  company: { name: 'edX', id: 2984 }
}

console.log(person.age);           // 25
console.log(person['company'].id) // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age); // 25  
console.log(person['company'].id) // 2984
```

# Objects

---

- JavaScript objects are used to store key-value pairs
- Values can be of any type, including arrays and objects!
- Values can be accessed by *myObject.property* or *myObject[ 'property' ]*

```
var person = {  
  name: 'John Doe',  
  age: 25,  
  isMale: true,  
  personality: ['patient', 'loyal', 'happy'],  
  company: { name: 'edX', id: 2984 }  
}  
  
console.log(person.age); // 25  
console.log(person['company'].id) // 2984
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {  
  name: 'Cooper',  
  type: 'dog'  
}  
  
console.log(pet.age);           // undefined  
pet.age = 11;  
console.log(pet.age);           // 11  
  
pet['status'] = 'good boy';  
console.log(pet.status);        // "good boy"
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {  
  name: 'Cooper',  
  type: 'dog'  
}  
  
console.log(pet.age);           // undefined  
pet.age = 11;  
console.log(pet.age);           // 11  
  
pet['status'] = 'good boy';  
console.log(pet.status);       // "good boy"
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {  
  name: 'Cooper',  
  type: 'dog'  
}  
  
console.log(pet.age);           // undefined  
pet.age = 11;  
console.log(pet.age);           // 11  
  
pet['status'] = 'good boy';  
console.log(pet.status);        // "good boy"
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {
  name: 'Cooper',
  type: 'dog'
}

console.log(pet.age);           // undefined
pet.age = 11;
console.log(pet.age);           // 11

pet['status'] = 'good boy';
console.log(pet.status);       // "good boy"
```



# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {  
  name: 'Cooper',  
  type: 'dog'  
}  
  
console.log(pet.age);           // undefined  
pet.age = 11;  
console.log(pet.age);         // 11  
  
pet['status'] = 'good boy';  
console.log(pet.status);       // "good boy"
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {  
  name: 'Cooper',  
  type: 'dog'  
}  
  
console.log(pet.age);           // undefined  
pet.age = 11;  
console.log(pet.age);           // 11  
  
pet['status'] = 'good boy';  
console.log(pet.status);        // "good boy"
```

# Modifying Objects

---

- Key-value pairs can be added to objects, even after their initial declaration

```
var pet = {
  name: 'Cooper',
  type: 'dog'
}

console.log(pet.age);           // undefined
pet.age = 11;
console.log(pet.age);           // 11

pet['status'] = 'good boy';
console.log(pet.status);       // "good boy"
```

# Summary

---

- JavaScript **arrays** let us create ordered collections of values with numeric indices
  
- JavaScript **objects** are collections of associated values with semantically meaningful names/keys



# Video 2.4

## Chris Murphy

# Conditional Statements

---

```
var a = . . .  
  
var b = . . .  
  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```

# Conditional Statements

---

```
var a = . . .  
  
var b = . . .  
  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```

# Conditional Statements

---

```
var a = . . .  
var b = . . .  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```



# Conditional Statements

---

```
var a = . . .  
  
var b = . . .  
  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```

# Conditional Statements

---

```
var a = . . .  
var b = . . .  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```

# Conditional Statements

---

```
var a = . . .  
var b = . . .  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max);
```

# Conditional Statements

---

```
var a = . . .  
var b = . . .  
var max; // undefined  
  
if (a > b) {  
    max = a;  
}  
else {  
    max = b;  
}  
  
console.log(max) ;
```

# Comparison and Logical Operators

## Comparison Operators

Operator	Description
==	equal to
===	equal to and same type
!=	not equal to
!==	not equal to or different type
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

# Comparison and Logical Operators

## Comparison Operators

Operator	Description
==	equal to
===	equal to and same type
!=	not equal to
!==	not equal to or different type
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

## Logical Operators

Operator	Description
	logical OR
&&	logical AND
!	logical NOT

# Double-equals vs. Triple-equals

---

- Use double-equals (==) when you only want to compare **values**

```
1 == '1' // true
```

# Double-equals vs. Triple-equals

---

- Use double-equals (==) when you only want to compare **values**

```
1 == '1' // true
```



# Double-equals vs. Triple-equals

---

- Use double-equals (==) when you only want to compare **values**
- Use triple-equals (===) when you want to compare values **and** type

```
1 == '1' // true
```

```
1 === '1' // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```



# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy

if (y === z) { . . . } // false! different types
```

# Comparing Truthy/Falsy Values

---

- Recall that any value can be used as a boolean
  - “Falsy” values: `null`, `undefined`, `0`, `NaN`, `''`
  - “Truthy” values: `'cow'`, `'false'`, `5`, etc...

```
var x; // undefined
if (x) { . . . } // false! undefined is falsy

x = 0;
if (x) { . . . } // false! 0 is falsy

x = 39;
if (x) { . . . } // true! 39 is truthy

var y = null;
var z; // undefined

if (y == z) { . . . } // true! falsy equals falsy
if (y === z) { . . . } // false! different types
```

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

```
5 < '20' // true
```

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

```
5 < '20' // true  
'5' < 20 // true
```

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

```
5 < '20' // true  
'5' < 20 // true
```

- Non-numeric strings are converted to NaN

```
5 > 'alligator' // false
```

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

```
5 < '20' // true
'5' < 20 // true
```

- Non-numeric strings are converted to NaN

```
5 > 'alligator' // false
5 < 'alligator' // also false!
```

# Comparing Numbers and Strings

---

- When comparing a string to a number, JavaScript will try to convert the string to a numeric form

```
5 < '20' // true
'5' < 20 // true
```

- Non-numeric strings are converted to NaN

```
5 > 'alligator' // false
5 < 'alligator' // also false!
```

- Non-numeric strings are compared alphabetically

```
'zebra' > 'giraffe' // true
```



# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Comparing Objects

---

- Objects are only considered equal if the variables are **aliases**, i.e. refer to the same object

```
var cooper = { age: 11 }  
var flanders = { age: 11 }  
  
if (cooper == flanders) { . . . } // false!  
  
var myDog = cooper;  
  
if (myDog == cooper) { . . . } // true!
```

# Loops

---

```
var n = ...  
var factorial = 1;
```

# Loops

---

```
var n = ...  
var factorial = 1;
```

```
for (var i = 1; i <= n; i++) {  
    factorial *= i;  
}
```



# Loops

---

```
var n = ...  
var factorial = 1;
```

```
for (var i = 1; i <= n; i++) {  
    factorial *= i;  
}
```

```
var i = 1;  
while (i <= n) {  
    factorial *= i;  
    i++;  
}
```

# Loops

---

```
var n = ...  
var factorial = 1;
```

```
for (var i = 1; i <= n; i++) {  
    factorial *= i;  
}
```

```
var i = 1;  
while (i <= n) {  
    factorial *= i;  
    i++;  
}
```

```
var i = 1;  
do {  
    factorial *= i;  
    i++;  
}  
while (i <= n);
```

# Summary

---

- JavaScript supports conditional statements and loops
- Comparison operators can be used to compare by value and also by type



# Video 2.5

## Chris Murphy

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
  var product = 1;  
  for (var i = 1; i <= n; i++) {  
    product *= i;  
  }  
  return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial (n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```



# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
  var product = 1;  
  for (var i = 1; i <= n; i++) {  
    product *= i;  
  }  
  return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
  var product = 1;  
  for (var i = 1; i <= n; i++) {  
    product *= i;  
  }  
  return product;  
}
```

```
var x = . . .
```

```
var f = factorial(x);
```

```
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Declaring and Using Functions

---

```
function factorial(n) {  
    var product = 1;  
    for (var i = 1; i <= n; i++) {  
        product *= i;  
    }  
    return product;  
}  
  
var x = . . .  
  
var f = factorial(x);  
  
console.log(f);
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```



# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

```
function isEven(n) {  
    return n % 2 == 0;  
}  
nums.every(isEven); // true
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

```
function isEven(n) {  
    return n % 2 == 0;  
}  
nums.every(isEven); // true
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

```
function isEven(n) {  
    return n % 2 == 0;  
}  
nums.every(isEven); // true
```

```
function square(n) {  
    return n * n;  
}  
var squares = nums.map(square); // [ 16, 64, 144, 4 ]
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
  console.log(n);  
}  
nums.forEach(print);
```

```
function isEven(n) {  
  return n % 2 == 0;  
}  
nums.every(isEven); // true
```

```
function square(n) {  
  return n * n;  
}  
var squares = nums.map(square); // [ 16, 64, 144, 4 ]
```

# Applying Functions to Arrays

---

```
var nums = [ 4, 8, 12, 2 ];
```

```
function print(n) {  
    console.log(n);  
}  
nums.forEach(print);
```

```
function isEven(n) {  
    return n % 2 == 0;  
}  
nums.every(isEven); // true
```

```
function square(n) {  
    return n * n;  
}  
var squares = nums.map(square); // [ 16, 64, 144, 4 ]
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them



# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

- Object arguments are passed by **reference**: the function **can** change them

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

- Object arguments are passed by **reference**: the function **can** change them

```
function changeMe(obj) {  
    obj.age++;  
}  
var p = { age: 30 };  
changeMe(p);  
console.log(p.age); // now 31
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

- Object arguments are passed by **reference**: the function **can** change them

```
function changeMe(obj) {  
    obj.age++;  
}  
var p = { age: 30 };  
changeMe(p);  
console.log(p.age); // now 31
```

# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

- Object arguments are passed by **reference**: the function **can** change them

```
function changeMe(obj) {  
    obj.age++;  
}  
var p = { age: 30 };  
changeMe(p);  
console.log(p.age); // now 31
```



# Pass-by-Value vs. Pass-by-Reference

---

- Primitive arguments are passed by **value**: the function cannot change them

```
function tryToChange(x) {  
    x = 4;  
}  
var y = 11;  
tryToChange(y);  
console.log(y); // still 11
```

- Object arguments are passed by **reference**: the function **can** change them

```
function changeMe(obj) {  
    obj.age++;  
}  
var p = { age: 30 };  
changeMe(p);  
console.log(p.age); // now 31
```

# Functions as Objects

---

- JavaScript functions are objects
  - Therefore, functions can take advantage of the benefits of an object, such as having properties
- Since JavaScript functions are objects, we can have variables refer to them

# Functions as Objects

---

- JavaScript functions are objects
  - Therefore, functions can take advantage of the benefits of an object, such as having properties
- Since JavaScript functions are objects, we can have variables refer to them

```
var add = function (a, b) {  
    return a + b;  
};  
  
console.log(add(3, 5));           // 8
```

# Functions as Objects

---

- JavaScript functions are objects
  - Therefore, functions can take advantage of the benefits of an object, such as having properties
- Since JavaScript functions are objects, we can have variables refer to them

```
var add = function (a, b) {  
    return a + b;  
};  
  
console.log(add(3, 5));           // 8
```

# Functions as Objects

---

- JavaScript functions are objects
  - Therefore, functions can take advantage of the benefits of an object, such as having properties
- Since JavaScript functions are objects, we can have variables refer to them

```
var add = function (a, b) {  
    return a + b;  
};  
  
console.log(add(3, 5));           // 8
```

# Functions as Objects

---

- JavaScript functions are objects
  - Therefore, functions can take advantage of the benefits of an object, such as having properties
- Since JavaScript functions are objects, we can have variables refer to them

```
var add = function (a, b) {  
    return a + b;  
};  
  
console.log(add(3, 5));           // 8
```

# Functions in Objects

---

- JavaScript functions can also be declared and used in objects

```
var johnDoe = {  
  name: 'John Doe',  
  age: '32',  
  greeting: function () {  
    return 'Hello! Nice Meeting You!';  
  }  
}  
  
console.log(johnDoe.greeting());
```

# Functions in Objects

---

- JavaScript functions can also be declared and used in objects

```
var johnDoe = {  
  name: 'John Doe',  
  age: '32',  
  greeting: function () {  
    return 'Hello! Nice Meeting You!';  
  }  
}  
  
console.log(johnDoe.greeting());
```



# Functions in Objects

---

- JavaScript functions can also be declared and used in objects

```
var johnDoe = {  
  name: 'John Doe',  
  age: '32',  
  greeting: function () {  
    return 'Hello! Nice Meeting You!';  
  }  
}  
  
console.log(johnDoe.greeting());
```

# Functions in Objects

---

- JavaScript functions can also be declared and used in objects

```
var johnDoe = {  
  name: 'John Doe',  
  age: '32',  
  greeting: function () {  
    return 'Hello! Nice Meeting You!';  
  }  
}  
  
console.log(johnDoe.greeting());
```

# Object Prototypes

---

- Every object in JavaScript has a **prototype**, accessed from the `__proto__` property in the object.
- The `__proto__` property is also an object, with its own `__proto__` property, and so on
- The root prototype of all objects is `Object.prototype`
- An object inherits the properties of its prototype

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```



# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Creating a Prototype

---

- Prototypes are created like any other JavaScript function or object
- The **this** keyword refers to the current object
- The **new** keyword can be used to create new objects from the same prototype

```
function Person (name, age) { // prototype
  this.name = name;
  this.age = age;
  this.greeting = function () {
    return 'Hello! My name is ' + this.name;
  }
}

var johnDoe = new Person('John Doe', 32);
johnDoe.greeting(); // Hello! My name is John Doe

var janeDoe = new Person('Jane Doe', 28);
janeDoe.greeting(); // Hello! My name is Jane Doe
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}  
  
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');  
  
sarahBrown.greeting();           //Hello! My name is Sarah Brown  
sarahBrown instanceof Person;    //true
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}  
  
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');  
  
sarahBrown.greeting();           //Hello! My name is Sarah Brown  
sarahBrown instanceof Person;    //true
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}  
  
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');  
  
sarahBrown.greeting();           //Hello! My name is Sarah Brown  
sarahBrown instanceof Person;    //true
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}  
  
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');  
  
sarahBrown.greeting();           //Hello! My name is Sarah Brown  
sarahBrown instanceof Person;    //true
```



# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}
```

```
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');
```

```
sarahBrown.greeting();           //Hello! My name is Sarah Brown  
sarahBrown instanceof Person;    //true
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {  
  this.__proto__ = new Person(name, age);  
  this.school = school;  
}  
  
var sarahBrown = new Student('Sarah Brown', 17, 'PennX');  
  
sarahBrown.greeting(); //Hello! My name is Sarah Brown  
sarahBrown instanceof Person; //true
```

# Extending a Prototype

---

- Prototypes can extend another prototype with more functionality
- To inherit a prototype, set the `__proto__` property of an object to the parent prototype

```
function Student (name, age, school) {
  this.__proto__ = new Person(name, age);
  this.school = school;
}

var sarahBrown = new Student('Sarah Brown', 17, 'PennX');

sarahBrown.greeting();           //Hello! My name is Sarah Brown
sarahBrown instanceof Person;   //true
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {  
  this.name = name;  
  this.age = age;  
  this.occupation = occupation;  
}
```

```
Person.prototype.planet = 'Earth';  
Person.prototype.introduction = function () {  
  return 'I am a ' + this.occupation;  
}
```

```
var johnDoe = new Person('John Doe', 32, 'Dentist');
```

```
johnDoe.planet;           //Earth  
johnDoe.introduction();  //I am a Dentist
```



# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Prototype Properties

---

- Properties and methods can be added to prototypes by adding them to the prototype property

```
var Person = function (name, age, occupation) {
  this.name = name;
  this.age = age;
  this.occupation = occupation;
}

Person.prototype.planet = 'Earth';
Person.prototype.introduction = function () {
  return 'I am a ' + this.occupation;
}

var johnDoe = new Person('John Doe', 32, 'Dentist');

johnDoe.planet;           //Earth
johnDoe.introduction();  //I am a Dentist
```

# Summary

---

- JavaScript supports functions
  - Primitives are passed by value
  - Objects are passed by reference
  
- Functions are objects and can be used to create objects
  
- JavaScript prototypes can be used to create “blueprints” for objects and can be modified dynamically



# Video 2.6

## Chris Murphy

# Review

---

- JavaScript strings are sequences of characters
- JavaScript strings are immutable
- Strings are objects and have their own functions

# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

```
var name = 'toucan';
```

# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

```
var name = 'toucan';  
  
name.length;           // 6
```



# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);      // 'c'
```

# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);      // 'c'
```

# Strings and Characters

---

- We can get the number of characters in a string using the `length` property
- We can access each character by its (0-based) index using `charAt` or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);        // 'c'  
name[3];              // 'c'
```

# Strings and Characters

---

- We can get the number of characters in a string using the **length** property
- We can access each character by its (0-based) index using **charAt** or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);        // 'c'  
name[3];               // 'c'
```

- Remember! JavaScript strings are immutable!

# Strings and Characters

---

- We can get the number of characters in a string using the **length** property
- We can access each character by its (0-based) index using **charAt** or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);        // 'c'  
name[3];               // 'c'
```

- Remember! JavaScript strings are immutable!

```
var animal = 'cat';
```

# Strings and Characters

---

- We can get the number of characters in a string using the **length** property
- We can access each character by its (0-based) index using **charAt** or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);        // 'c'  
name[3];               // 'c'
```

- Remember! JavaScript strings are immutable!

```
var animal = 'cat';  
  
animal[0] = 'r';
```

# Strings and Characters

---

- We can get the number of characters in a string using the **length** property
- We can access each character by its (0-based) index using **charAt** or array notation

```
var name = 'toucan';  
  
name.length;           // 6  
  
name.charAt(3);        // 'c'  
name[3];               // 'c'
```

- Remember! JavaScript strings are immutable!

```
var animal = 'cat';  
  
animal[0] = 'r';  
  
console.log(animal); // still 'cat'
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)



# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'  
console.log(friend); // 'turtle'
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'  
console.log(friend); // 'turtle'
```

```
var message = '_hello everyone_' ;
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'  
console.log(friend); // 'turtle'
```

```
var message = ' hello everyone '  
message = message.trim(); // 'hello everyone'
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'  
console.log(friend); // 'turtle'
```

```
var message = ' hello everyone '  
message = message.trim(); // 'hello everyone'
```

```
var myAnimal = 'cat'.concat('mouse');
```

# Modifying Strings

---

- We can modify a string but these functions return a **new** string (since strings are immutable!)

```
var friend = 'turtle';  
  
friend.toUpperCase(); // 'TURTLE'  
console.log(friend); // 'turtle'
```

```
var message = ' hello everyone ';  
message = message.trim(); // 'hello everyone'
```

```
var myAnimal = 'cat'.concat('mouse');  
console.log(myAnimal); // 'catmouse'
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```



# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```



# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Searching Strings

---

- We can determine whether a string starts with, ends with, or includes other strings

```
var msg = 'programming in JavaScript is fun';  
  
msg.startsWith('programming'); // true  
msg.startsWith('PROGRAMMING'); // false  
  
msg.endsWith('is fun'); // true  
  
msg.includes('JavaScript'); // true
```

- We can also get the starting index of a contained substring

```
var title = 'the title of my book';  
var start = title.search('title'); // 4  
start = title.search('banana'); // -1
```

# Regular Expressions

---

- A **regular expression** is a pattern of characters
- A string **matches** a regular expression if it adheres to the same pattern
- Example: “consists of exactly three digits (0-9)”
  - ‘123’ matches
  - ‘abc’ does not match
  - ‘12’ does not match
  - ‘12345’ does not match

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression



# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/);
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/);
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1  
  
status.search(/very/i);
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's `search` function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1  
  
status.search(/very/i); // 13
```



# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1  
  
status.search(/very/i); // 13
```

- Or, we can use the regex's **test** function

```
/script/.test('javascript is so much fun!'); // true
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1  
  
status.search(/very/i); // 13
```

- Or, we can use the regex's **test** function

```
/script/.test('javascript is so much fun!'); // true
```

# Simple Regular Expression Matching

---

- We can pass a regular expression to the string's **search** function to see if it matches the pattern
- In general, it is considered a match if **any** part of the string matches the regular expression

```
var status = 'I am working VERY hard';  
  
status.search(/VERY/); // 13  
  
status.search(/very/); // -1  
  
status.search(/very/i); // 13
```

- Or, we can use the regex's **test** function

```
/script/.test('javascript is so much fun!'); // true
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/ [012] /);
```



# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/ [012] /);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/ [012] /); // 4
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);             // true
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);             // true
```

- Or ranges of characters or special characters

```
var password = 'password4real';
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);           // true
```

- Or ranges of characters or special characters

```
var password = 'password4real';  
password.search(/[a-z]/);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);             // true
```

- Or ranges of characters or special characters

```
var password = 'password4real';  
password.search(/[a-z]/);           // 0
```



# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);           // true
```

- Or ranges of characters or special characters

```
var password = 'password4real';  
password.search(/[a-z]/);         // 0  
password.search(/\d/);
```

# Specifying Ranges of Characters

---

- We can also specify multiple valid characters that we want to consider for matching
- For instance, we can look for specific characters

```
var numbers = '5 8 2 5 7 6';  
numbers.search(/[012]/);           // 4  
/[012]/.test(numbers);             // true
```

- Or ranges of characters or special characters

```
var password = 'password4real';  
password.search(/[a-z]/);           // 0  
password.search(/\d/);               // 8
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/ [0-9] [a-z] [0-9] /);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9] [a-z] [0-9]/);
```



# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/ [0-9] [a-z] [0-9] /);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

- Or look for characters **not** in a range

```
var chars = 'abc123K456';  
chars.search(/[^0-9a-z]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

- Or look for characters **not** in a range

```
var chars = 'abc123K456';  
chars.search(/[^0-9a-z]/);
```



# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

- Or look for characters **not** in a range

```
var chars = 'abc123K456';  
chars.search(/^[^0-9a-z]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

- Or look for characters **not** in a range

```
var chars = 'abc123K456';  
chars.search(/[^0-9a-z]/);
```

# Using Ranges

---

- We can combine different ranges

```
var code = 'abc123d4e5';  
code.search(/[0-9][a-z][0-9]/); // 5
```

- Or look for characters **not** in a range

```
var chars = 'abc123k456';  
chars.search(/[^0-9a-z]/); // 6
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b');
```

```
/[a-z][0-9]?[a-z]/.test('abc');
```

```
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b');
```

```
/[a-z][0-9]?[a-z]/.test('abc');
```

```
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b');
```

```
/[a-z][0-9]?[a-z]/.test('abc');
```

```
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b');
```

```
/[a-z][0-9]?[a-z]/.test('abc');
```

```
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b');
```

```
/[a-z][0-9]?[a-z]/.test('abc');
```

```
/[a-z][0-9]?[a-z]/.test('a123b');
```



# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc');  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc');  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
/[a-z][0-9]?[a-z]/.test('abc');  
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc');  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc');  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b'); // true  
/[a-z][0-9]?[a-z]/.test('abc'); // true  
/[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b');
```



# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b'); // true  
/[a-z][0-9]?[a-z]/.test('abc'); // true  
/[a-z][0-9]?[a-z]/.test('a123b'); // false
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
/[a-z][0-9]?[a-z]/.test('a1b'); // true  
/[a-z][0-9]?[a-z]/.test('abc'); // true  
/[a-z][0-9]?[a-z]/.test('a123b');// false
```

- Or optional **multiple** occurrences

```
/[a-z][0-9]*[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b');
```



# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b');
```

# Quantifiers

---

- We may want to know whether the string contains an optional **single** occurrence

```
[a-z][0-9]?[a-z]/.test('a1b'); // true  
[a-z][0-9]?[a-z]/.test('abc'); // true  
[a-z][0-9]?[a-z]/.test('a123b'); // false
```

- Or optional **multiple** occurrences

```
[a-z][0-9]*[a-z]/.test('a123b'); // true
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');  
 /^[a-z][0-9]/.test('ab12');  
  
/[a-z][a-z]$/.test('123abc');  
/[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');  
 /^[a-z][0-9]/.test('ab12');  
  
/[a-z][a-z]$/.test('123abc');  
/[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^ [a-z] [0-9] /.test ('a1b');  
/^ [a-z] [0-9] /.test ('ab12');  
  
/[a-z] [a-z] $/.test ('123abc');  
/[a-z] [a-z] $/.test ('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^ [a-z] [0-9] /.test ('a1b');  
/^ [a-z] [0-9] /.test ('ab12');  
  
/[a-z] [a-z] $/.test ('123abc');  
/[a-z] [a-z] $/.test ('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^ [a-z] [0-9] /.test ('a1b');           // true
/^ [a-z] [0-9] /.test ('ab12');

/[a-z] [a-z] $/.test ('123abc');
/[a-z] [a-z] $/.test ('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true  
 /^[a-z][0-9]/.test('ab12');  
  
/[a-z][a-z]$/.test('123abc');  
/[a-z][a-z]$/.test('123abc456');
```



# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');

 /[a-z][a-z]$/.test('123abc');
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');

 /[a-z][a-z]$/.test('123abc');
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/ .test('123abc');
 /[a-z][a-z]$/ .test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');      // true
 /[a-z][a-z]$/.test('123abc456');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');
```



# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');    // false
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');
 /^[a-z][0-9][a-z]$/.test('a1b2c');
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');
 /^[a-z][0-9][a-z]$/.test('a1b2c');
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b');
/^[a-z][0-9][a-z]$/.test('a1b2c');
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');
 /^[a-z][0-9][a-z]$/.test('a1b2c');
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9] [a-z]$/.test('a1b');
/^[a-z][0-9][a-z]$/.test('a1b2c');
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b');           // true
/^[a-z][0-9]/.test('ab12');           // false

/[a-z][a-z]$/.test('123abc');         // true
/[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b');      // true
/^[a-z][0-9][a-z]$/.test('a1b2c');
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```



# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b'); // true
/^[a-z][0-9][a-z]$/.test('a1b2c');
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b');           // true
/^[a-z][0-9]/.test('ab12');          // false

/[a-z][a-z]$/.test('123abc');       // true
/[a-z][a-z]$/.test('123abc456');    // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b');     // true
/^[a-z][0-9][a-z]$/.test('a1b2c');  // false
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c'); // true
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');     // true
 /^[a-z][0-9][a-z]$/.test('a1b2c');   // false
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c'); // true
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');     // true
 /^[a-z][0-9][a-z]$/.test('a1b2c');   // false
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c'); // true
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b');           // true
/^[a-z][0-9]/.test('ab12');          // false

/[a-z][a-z]$/.test('123abc');        // true
/[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b');      // true
/^[a-z][0-9][a-z]$.test('a1b2c');    // false
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c'); // true
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');     // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');     // true
 /^[a-z][0-9][a-z]$/.test('a1b2c');  // false
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b'); // true
/^[a-z][0-9][a-z]$/.test('a1b2c'); // false

/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^ [a-z] [0-9] /.test ('a1b'); // true
/^ [a-z] [0-9] /.test ('ab12'); // false

/[a-z] [a-z] $/.test ('123abc'); // true
/[a-z] [a-z] $/.test ('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^ [a-z] [0-9] [a-z] $/.test ('a1b'); // true
/^ [a-z] [0-9] [a-z] $/.test ('a1b2c'); // false

/^ [a-z] [0-9a-z]* [a-z] $/.test ('a1b2c'); // true
```



# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^ [a-z] [0-9] /.test ('a1b');           // true
/^ [a-z] [0-9] /.test ('ab12');          // false

/[a-z] [a-z] $/.test ('123abc');         // true
/[a-z] [a-z] $/.test ('123abc456');     // false
```

- This lets us detect **exact** matches

```
/^ [a-z] [0-9] [a-z] $/.test ('a1b');    // true
/^ [a-z] [0-9] [a-z] $/.test ('a1b2c');  // false
/^ [a-z] [0-9a-z]* [a-z] $/.test ('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
 /^[a-z][0-9]/.test('a1b');           // true
 /^[a-z][0-9]/.test('ab12');          // false

 /[a-z][a-z]$/.test('123abc');        // true
 /[a-z][a-z]$/.test('123abc456');    // false
```

- This lets us detect **exact** matches

```
 /^[a-z][0-9][a-z]$/.test('a1b');     // true
 /^[a-z][0-9][a-z]$/.test('a1b2c');   // false
 /^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b'); // true
/^[a-z][0-9][a-z]$/.test('a1b2c'); // false
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b'); // true
/^[a-z][0-9][a-z]$/.test('a1b2c'); // false
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c');
```

# startsWith and endsWith Matches

---

- Regular expressions can tell us if a string **contains** a pattern, but we may want to know if the string **starts** or **ends** with the pattern

```
/^[a-z][0-9]/.test('a1b'); // true
/^[a-z][0-9]/.test('ab12'); // false

/[a-z][a-z]$/.test('123abc'); // true
/[a-z][a-z]$/.test('123abc456'); // false
```

- This lets us detect **exact** matches

```
/^[a-z][0-9][a-z]$/.test('a1b'); // true
/^[a-z][0-9][a-z]$/.test('a1b2c'); // false
/^[a-z][0-9a-z]*[a-z]$/.test('a1b2c'); // true
```

# Summary

---

- JavaScript strings are immutable but provide functions that allow us to create new, modified versions of them
- Strings have **startsWith**, **endsWith**, **includes**, and **search** functions
- We can also use regular expressions' **test** function to check for matches in a string



# Video 2.7

## Chris Murphy

# How do we use JavaScript and HTML?

---

- We motivated this part of the course by saying that we wanted a way to dynamically generate HTML
- Now that we've seen JavaScript, how can we use it to access/modify HTML elements?
- This is done by using the **DOM**



# What is the DOM?

---

- The **D**ocument **O**bject **M**odel is a structured tree representation of a web page
- The HTML of every web page is turned into a DOM representation by the browser

# What does the DOM look like?

---

- HTML

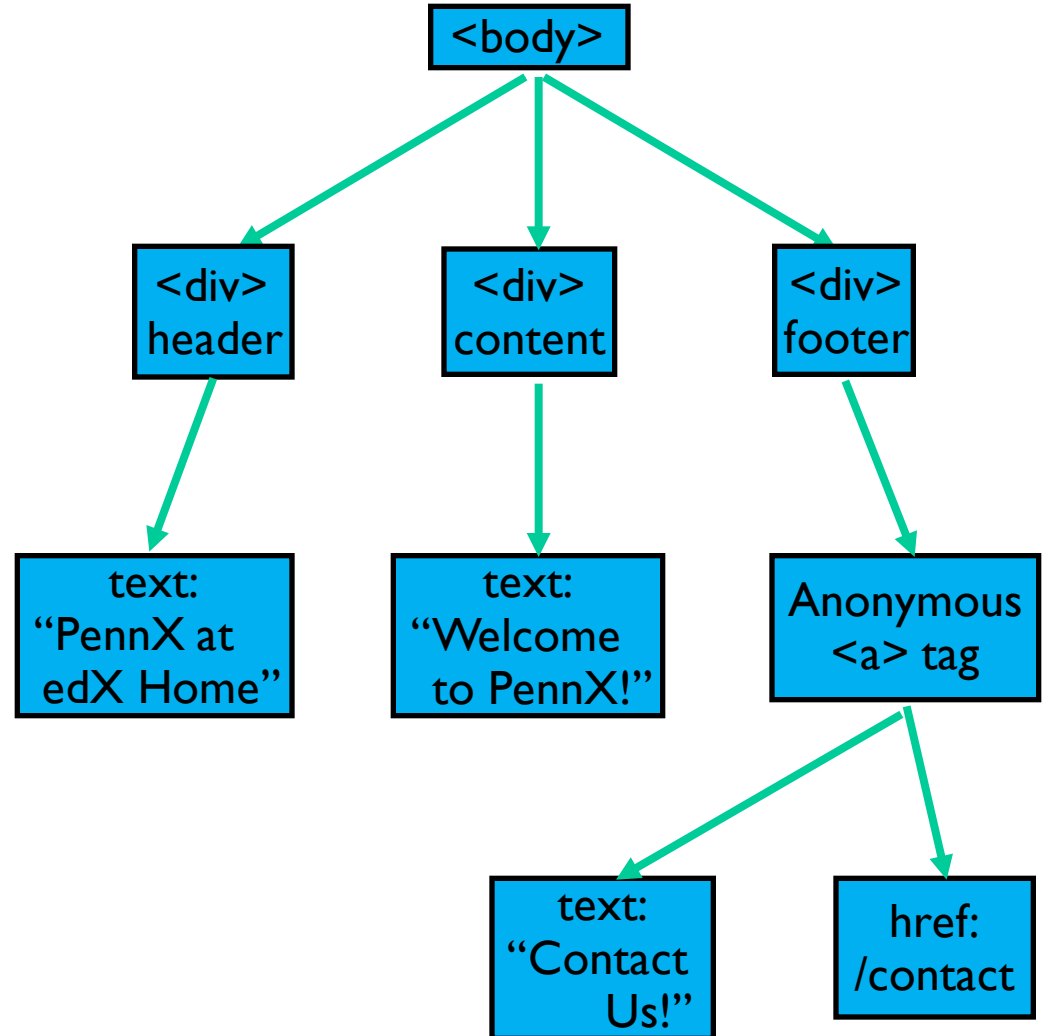
```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </html>
```

# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree

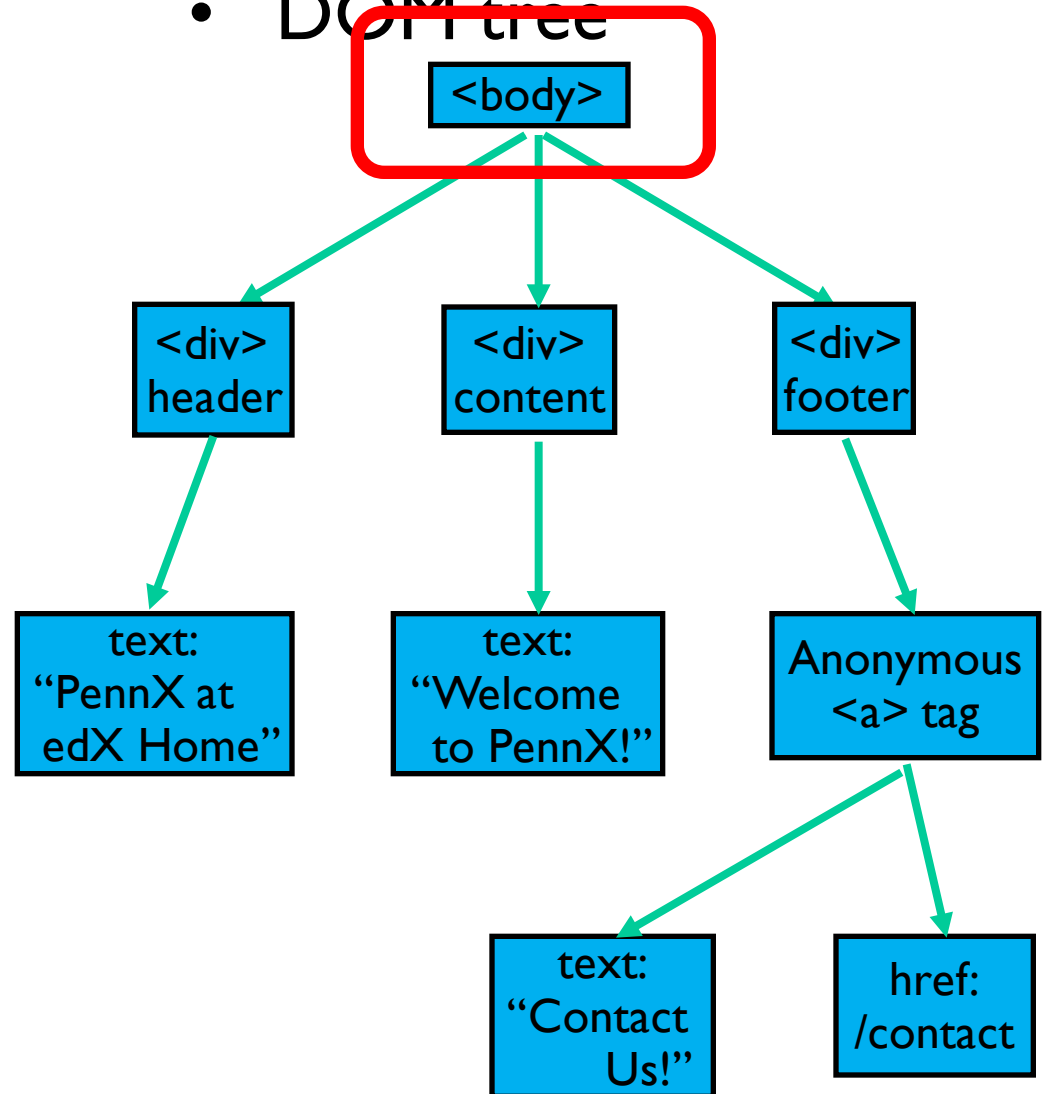


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree

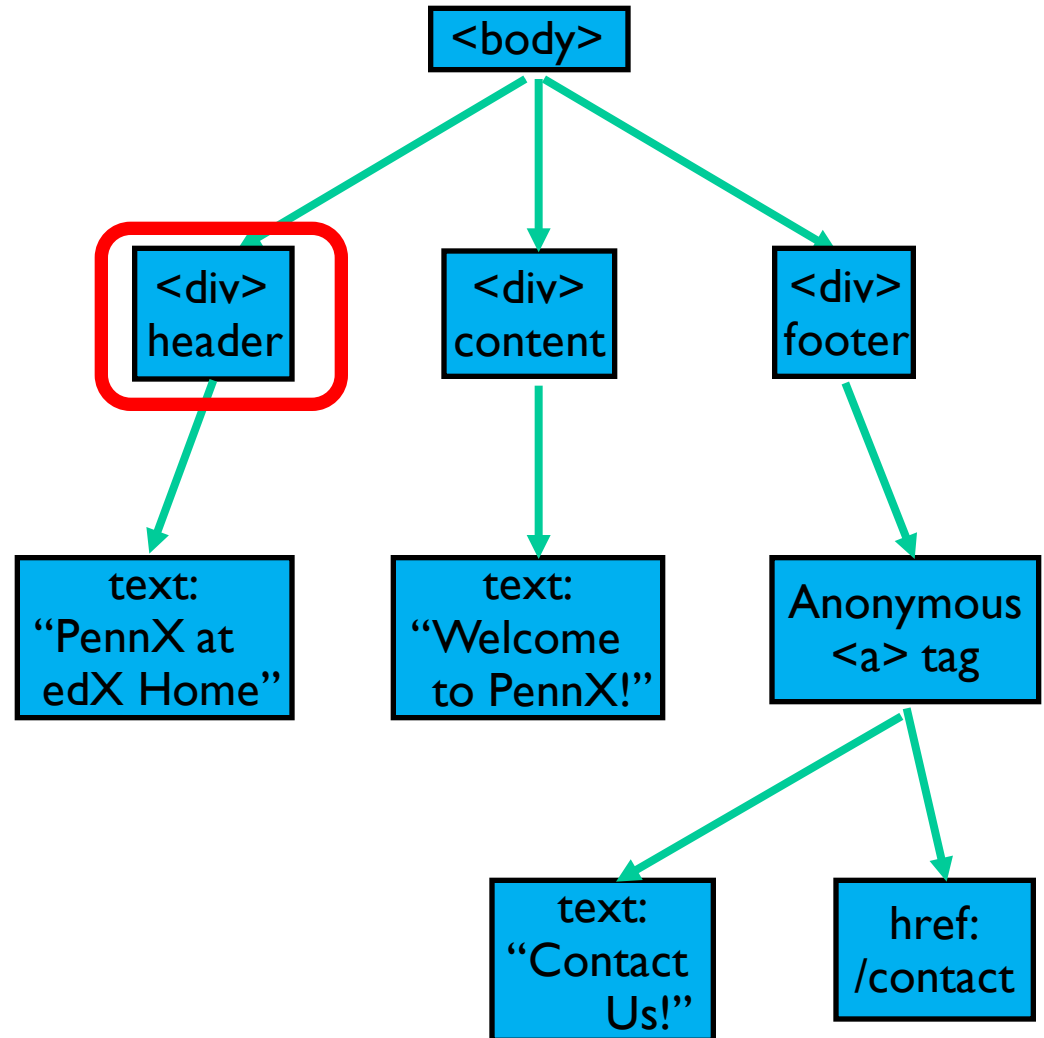


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree

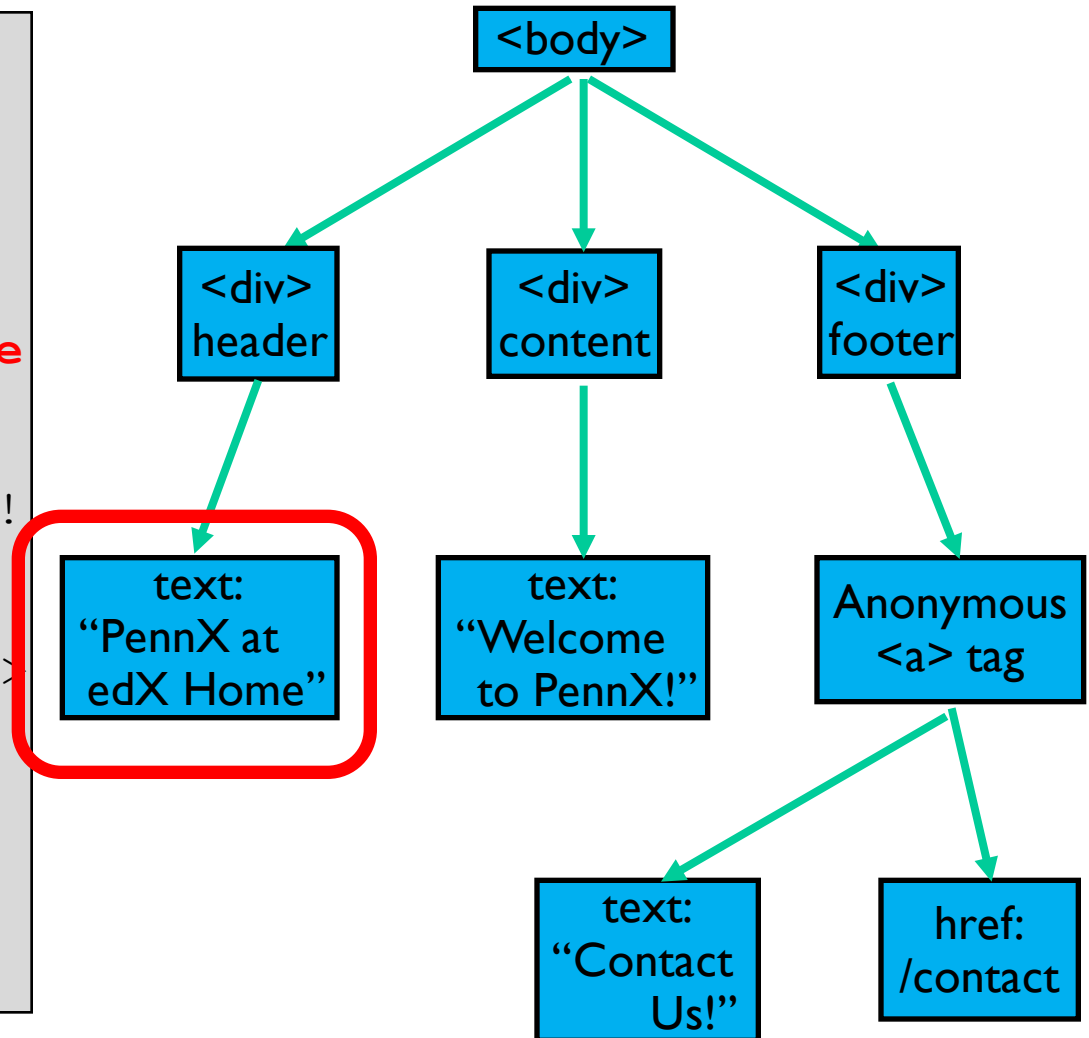


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree

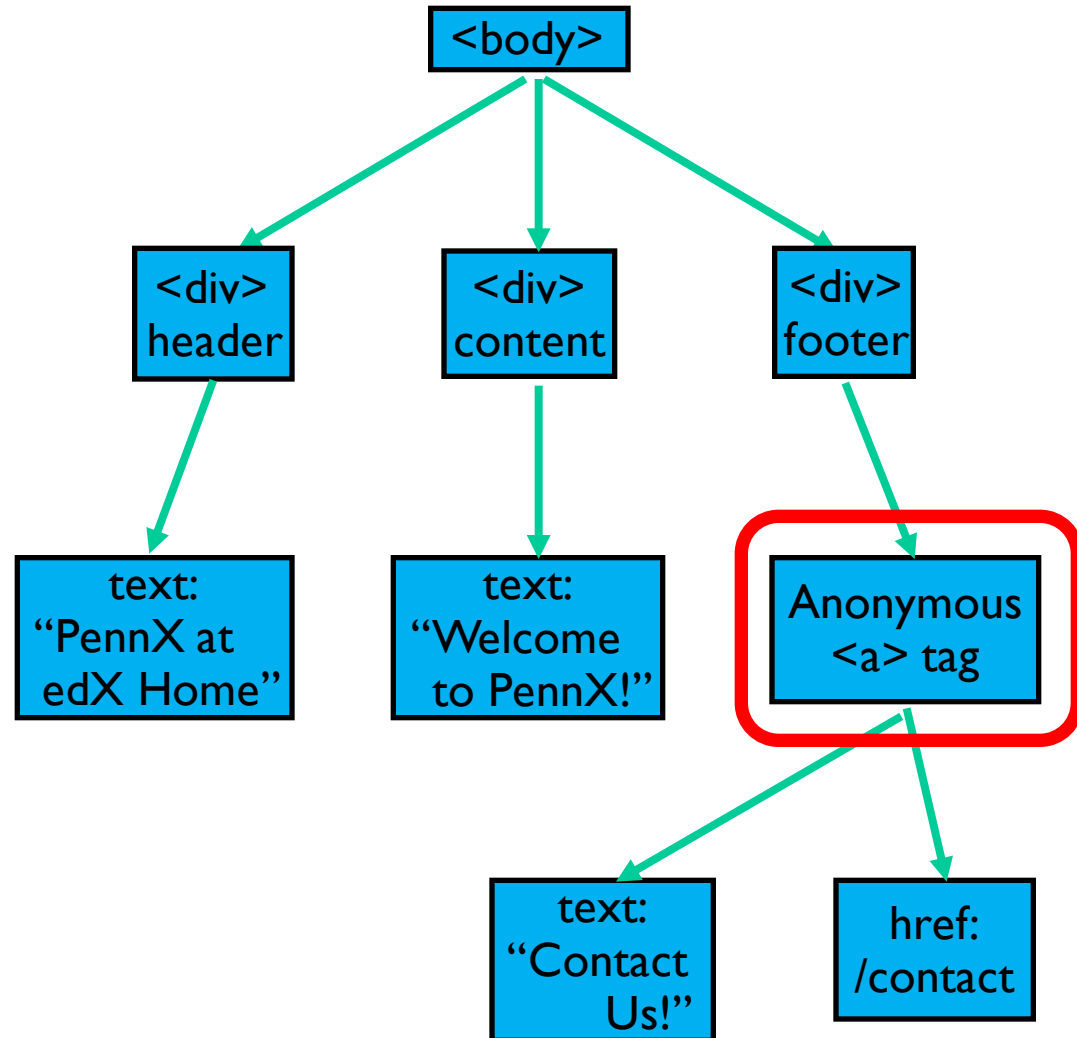


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree

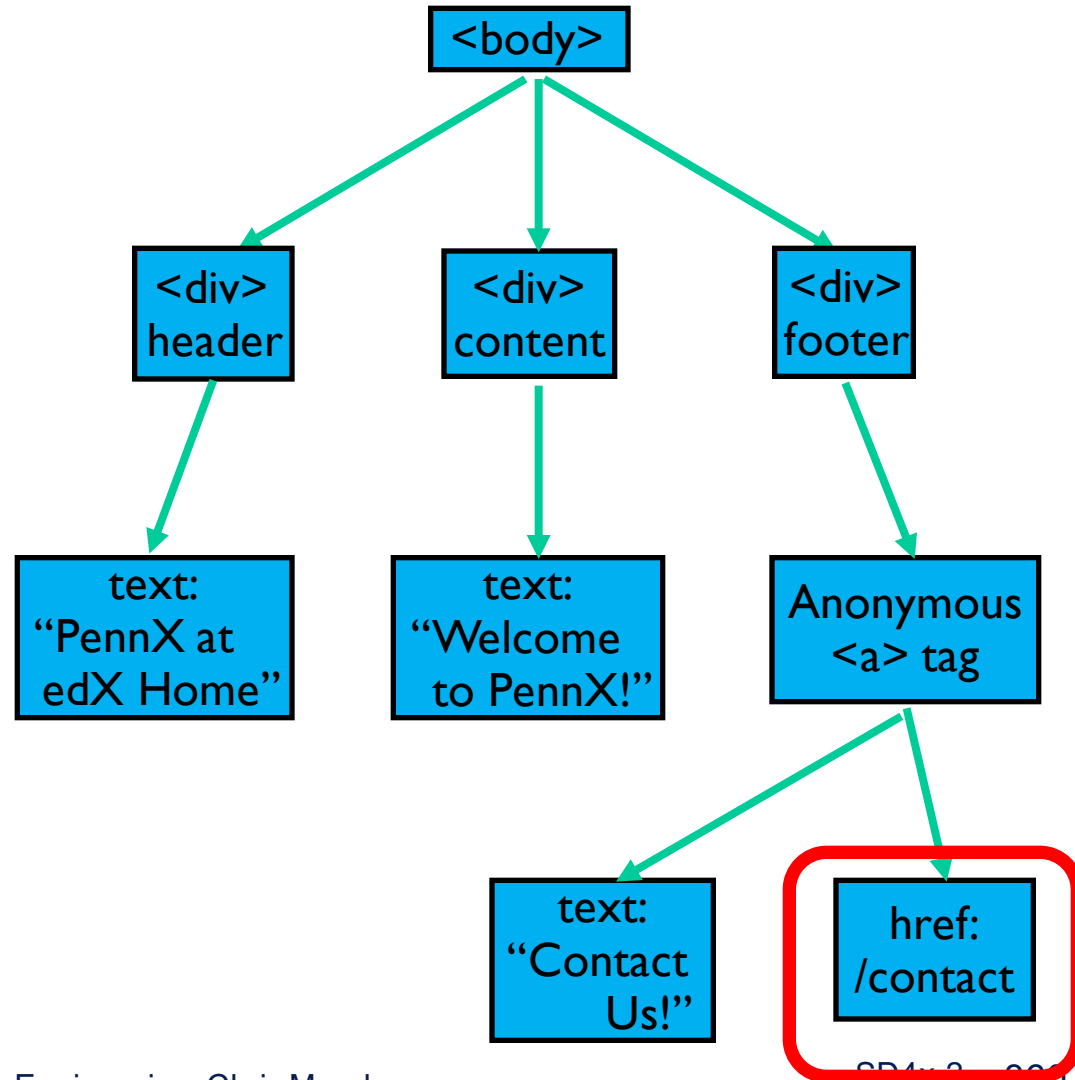


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree



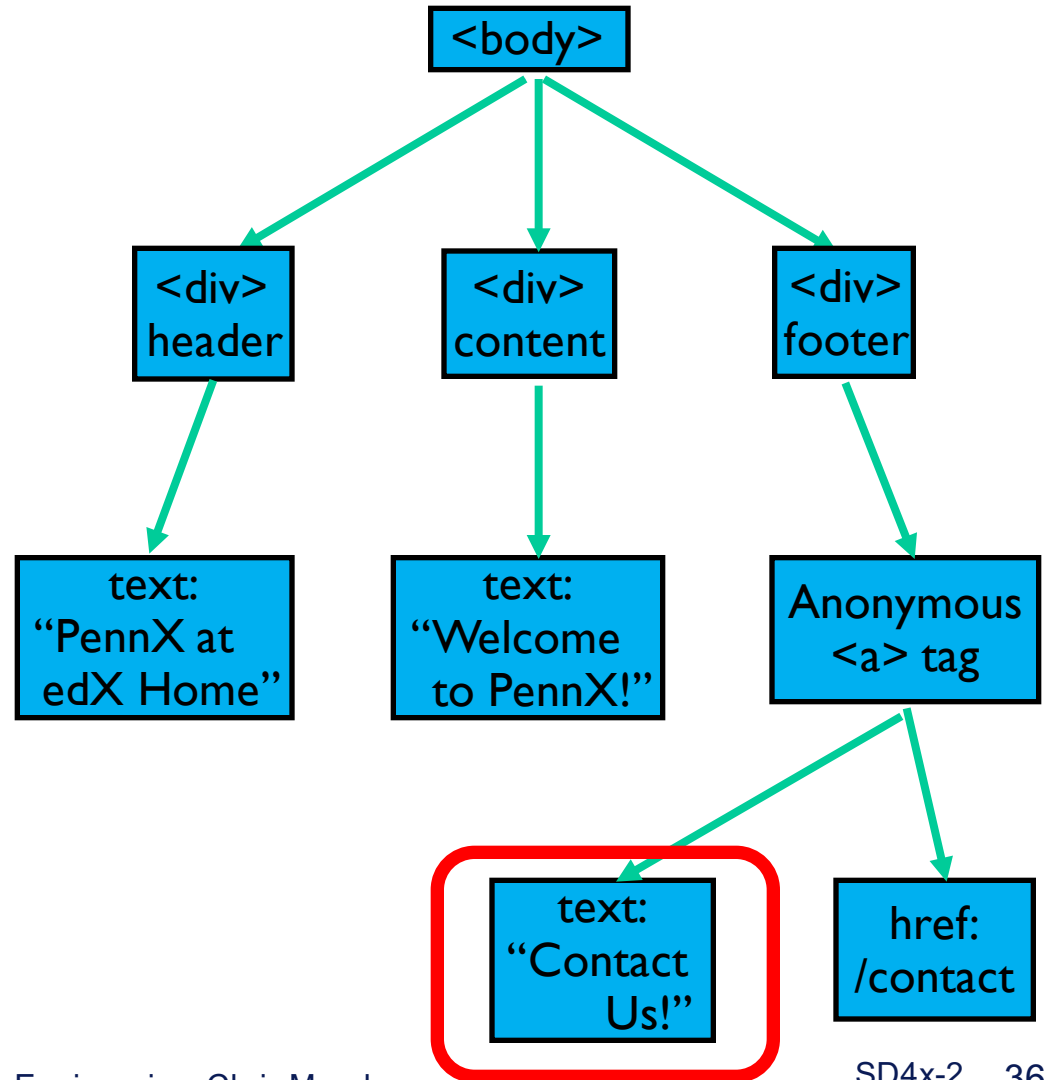


# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

- DOM tree



# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </html>
```

- DOM in console

```
<!DOCTYPE html>
...<html> == $0
  <head></head>
  ▼ <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    ▼ <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </html>
```

- DOM in console

```
<!DOCTYPE html>
...<html> == $0
  <head></head>
  ▼<body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    ▼<div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```

# What does the DOM look like?

- HTML

```
<!DOCTYPE html>
<html>
  <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </html>
```

- DOM in console

```
<!DOCTYPE html>
...<html> == $0
  <head></head>
  ▼ <body>
    <div id="header">
      PennX at edX Home
    </div>
    <div id="content">
      Welcome to PennX!
    </div>
    ▼ <div id="footer">
      <a href="/contact">
        Contact Us!
      </a>
    </div>
  </body>
</html>
```



# Why the DOM?

---

- Remember, HTML specifies the **structure** of the content on the Web page
- The DOM provides a way for us to programmatically access that structure in JavaScript

# DOM Example

---

- The root DOM object can be accessed by the object called `document`
- Elements in this DOM tree can be retrieved and manipulated

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```



# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is  </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```

# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date();
</script>

</body>
</html>
```



# DOM Example

---

- The root DOM object can be accessed by the object called **document**
- Elements in this DOM tree can be retrieved and manipulated

```
<html>
<body>

<div>
  The current date/time is <span id="dateTime"> </span>.
</div>

<script>
  var dateTimeField = document.getElementById('dateTime');
  dateTimeField.innerHTML = new Date() ;
</script>

</body>
</html>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using `localStorage`

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using `localStorage`

```
<div>
  You have visited this page   times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited) ;
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```



# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using **localStorage**

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# DOM Example

---

- Data can be stored in the browser across multiple page requests using `localStorage`

```
<div>
  You have visited this page <span id="report"> </span> times.
</div>

<script>
  var timesVisited = 0;
  if (localStorage.timesVisited) {
    timesVisited = parseInt(localStorage.timesVisited);
  }
  timesVisited += 1;
  localStorage.setItem('timesVisited', timesVisited);

  var report = document.getElementById('report');
  report.innerHTML = timesVisited;

  if (timesVisited > 10)
    report.style.backgroundColor = 'red';
</script>
```

# Objects as JSON

---

- JSON = JavaScript **O**bject **N**otation
- JSON is a **textual** representation of a JavaScript Object that can be stored as a string, in a .json file, or be exchanged between programs
- A sample JSON file or string might look like this:

```
{  
  "name": "John Doe",  
  "age": 25,  
  "isMale": true,  
  "personality": ["patient", "loyal", "happy"],  
  "company": { "name": "EdX", "id": 2984 }  
}
```

# Converting between JSON and Objects

---

- JavaScript objects can be converted to a JSON string via `JSON.stringify(myObject)`
- String representations can be converted back to an object via `JSON.parse(jsonString)`
- All values must be a string, number, array, boolean, null, or another valid JSON object

# Storing JSON

---

- A great application of JSON usage is to store JSON strings in local browser storage:

```
localStorage.myJSON = JSON.stringify(myObject);  
  
// ... in a later session  
myObject = JSON.parse(localStorage.myJSON);
```

- Later on, as you learn about server-side JavaScript, you will also learn how to use JSON data to communicate with a server or API.

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```



# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

`<script>`

**`var timesVisited = 0;`**

`var lastVisitDate = 'never';`

`if (localStorage.lastVisit) {`

`var lastVisit = JSON.parse(localStorage.lastVisit);`

`timesVisited = lastVisit.numVisits;`

`lastVisitDate = lastVisit.date;`

`}`

`document.getElementById('lastVisitDate').innerHTML = lastVisitDate;`

`timesVisited++;`

`document.getElementById('report').innerHTML = timesVisited;`

`var myLastVisit = { }`

`myLastVisit.date = new Date();`

`myLastVisit.numVisits = timesVisited;`

`localStorage.lastVisit = JSON.stringify(myLastVisit);`

`</script>`

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

`<script>`

`var timesVisited = 0;`

`var lastVisitDate = 'never';`

`if (localStorage.lastVisit) {`

`var lastVisit = JSON.parse(localStorage.lastVisit);`

`timesVisited = lastVisit.numVisits;`

`lastVisitDate = lastVisit.date;`

`}`

`document.getElementById('lastVisitDate').innerHTML = lastVisitDate;`

`timesVisited++;`

`document.getElementById('report').innerHTML = timesVisited;`

`var myLastVisit = { }`

`myLastVisit.date = new Date();`

`myLastVisit.numVisits = timesVisited;`

`localStorage.lastVisit = JSON.stringify(myLastVisit);`

`</script>`

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

`<script>`

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```



# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# DOM and JSON

You have accessed this page `<span id="report"></span>` times.

`<p>`

Your last visit was `<span id="lastVisitDate"></span>`.

```
<script>
```

```
var timesVisited = 0;
```

```
var lastVisitDate = 'never';
```

```
if (localStorage.lastVisit) {
```

```
    var lastVisit = JSON.parse(localStorage.lastVisit);
```

```
    timesVisited = lastVisit.numVisits;
```

```
    lastVisitDate = lastVisit.date;
```

```
}
```

```
document.getElementById('lastVisitDate').innerHTML = lastVisitDate;
```

```
timesVisited++;
```

```
document.getElementById('report').innerHTML = timesVisited;
```

```
var myLastVisit = { }
```

```
myLastVisit.date = new Date();
```

```
myLastVisit.numVisits = timesVisited;
```

```
localStorage.lastVisit = JSON.stringify(myLastVisit);
```

```
</script>
```

# Summary

---

- JavaScript can use the DOM to retrieve/modify HTML elements
  - `document.getElementById( 'id' )` returns the specific HTML element with that ID
  - `element.innerHTML` can be modified to change the element's HTML/content
  - `element.style` can be modified to change the element's CSS/appearance
- We can use `localStorage` to save values across page requests
- Objects can be converted to string representations known as JSON





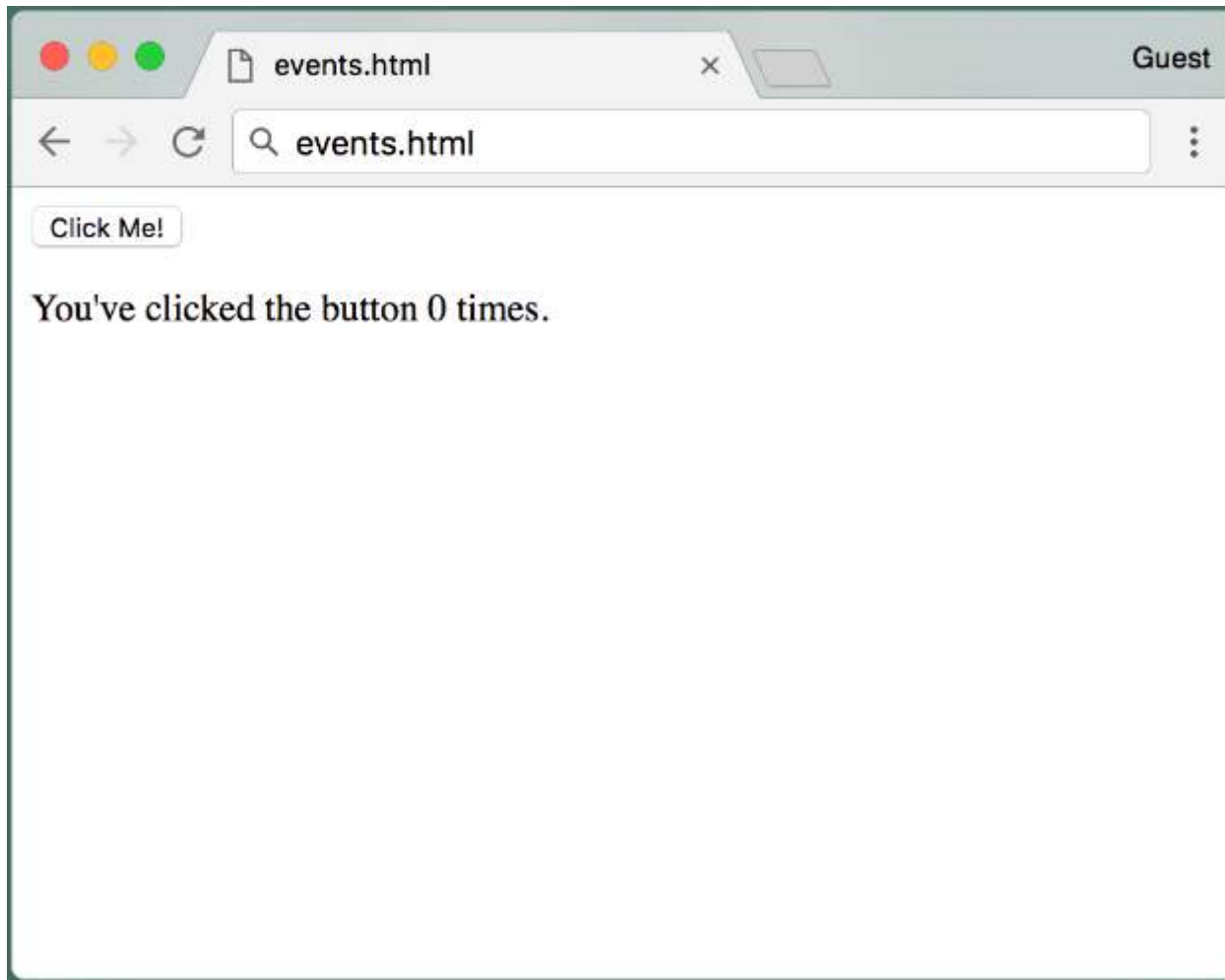
# Video 2.8

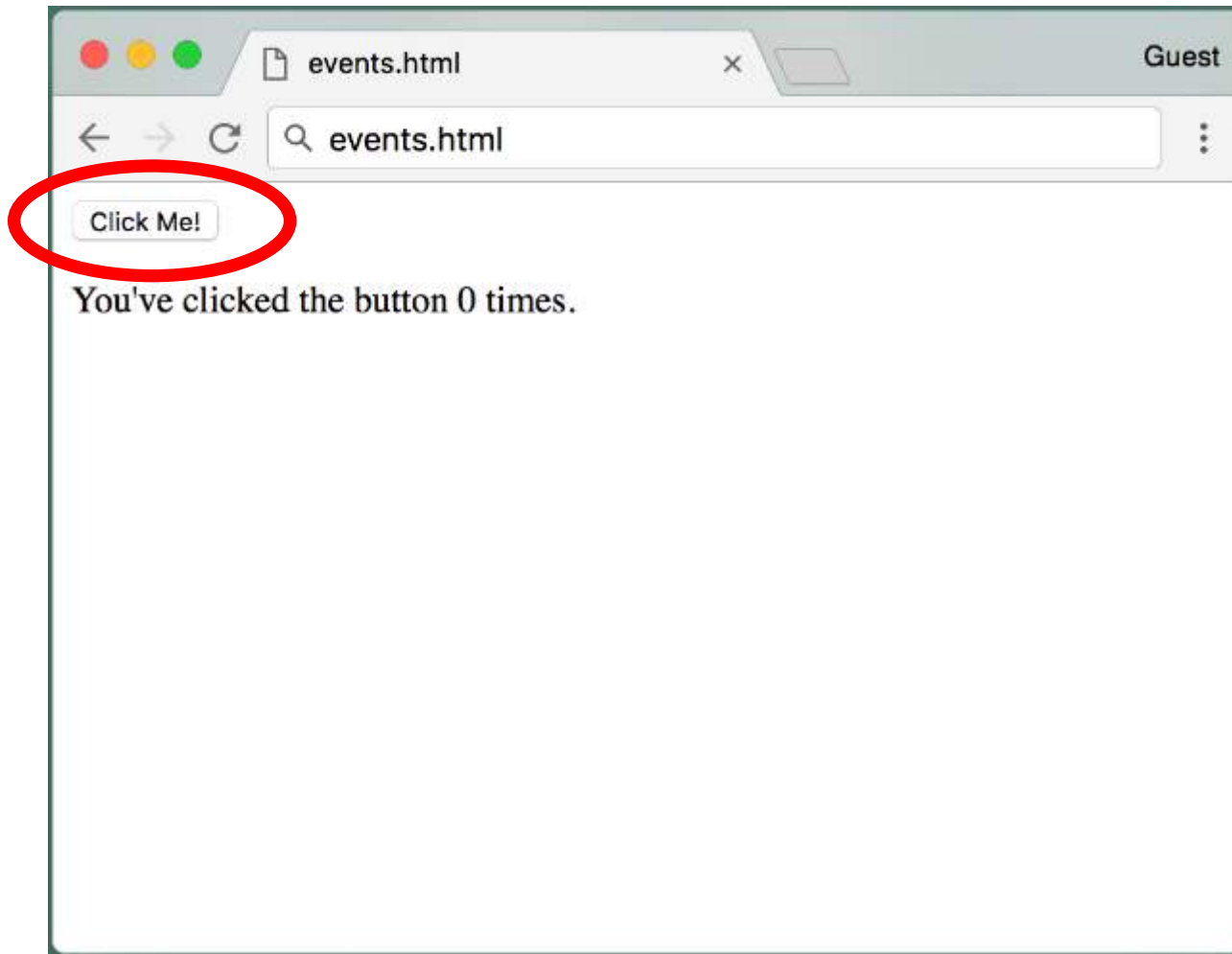
## Chris Murphy

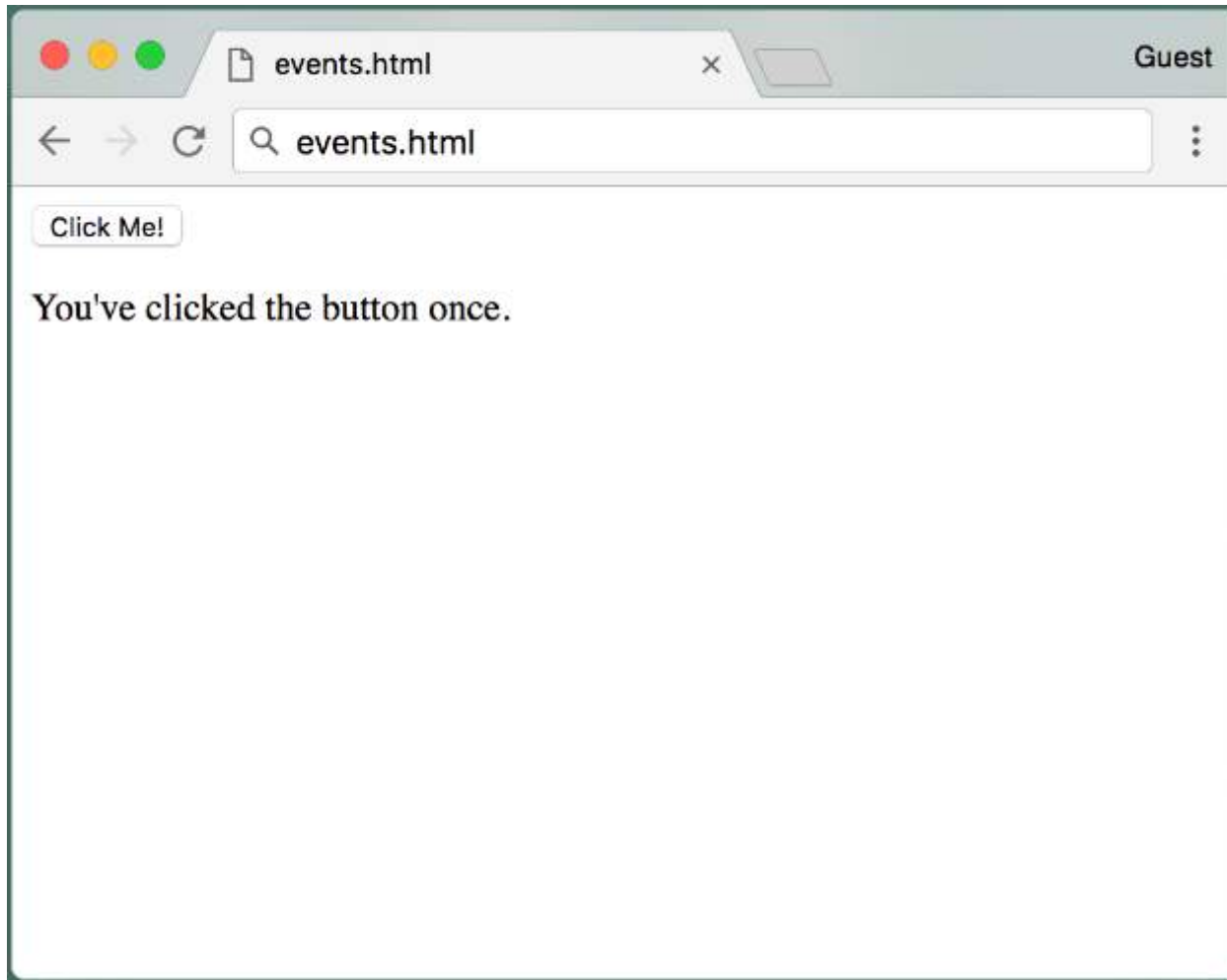
# Review: HTML, JavaScript, DOM

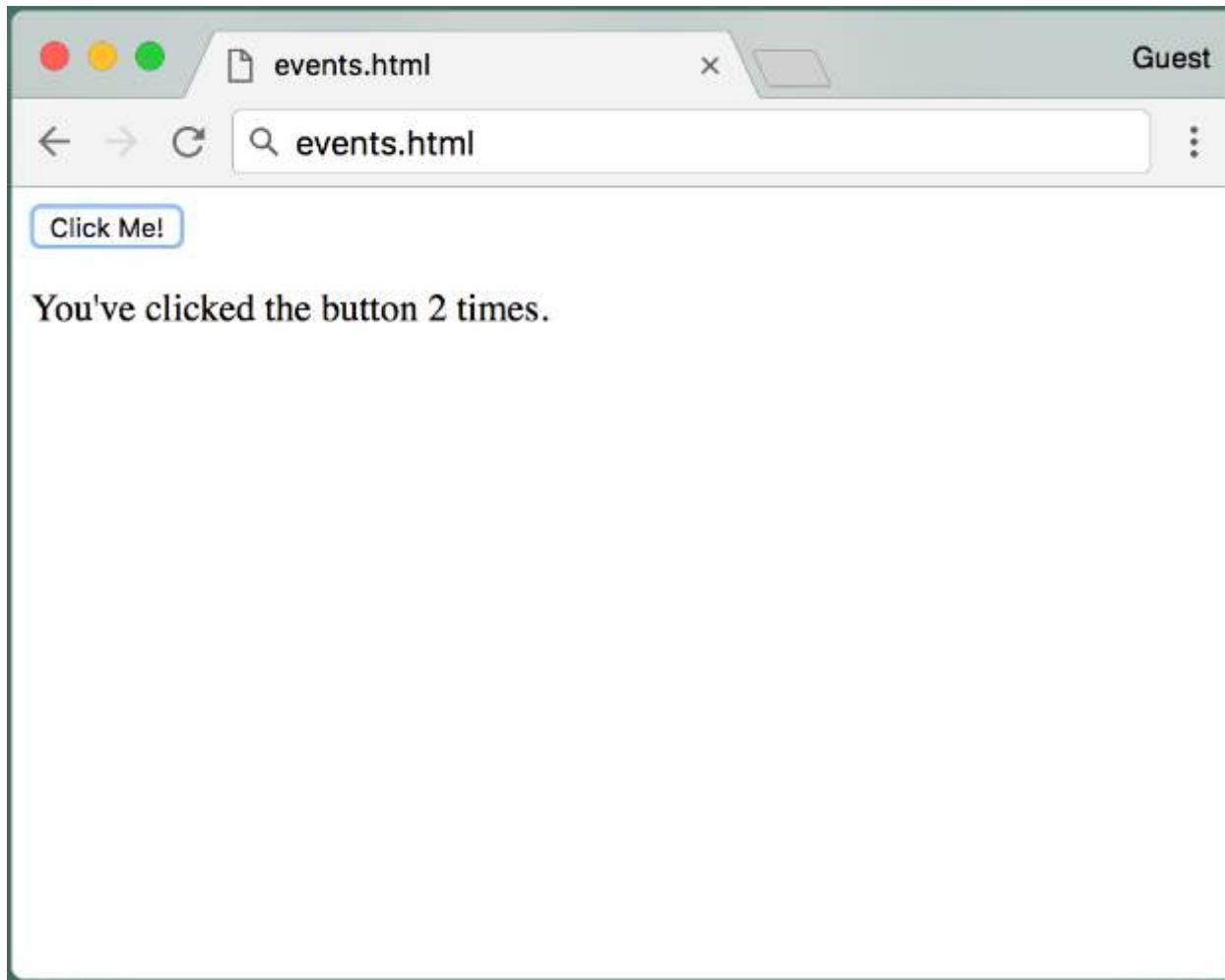
---

- Previously we saw that JavaScript can use the DOM to retrieve/modify HTML elements
  - `document.getElementById('id')` returns the specific HTML element with that ID
  - `element.innerHTML` can be modified to change the element's HTML/content
  - `element.style` can be modified to change the element's CSS/appearance
- How can we do this in response to user events?









```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```



```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
  clicks++;
  var numClicksSpan = document.getElementById('numClicks');
  if (clicks == 1)
    numClicksSpan.innerHTML = 'once';
  else
    numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
  clicks++;
  var numClicksSpan = document.getElementById('numClicks');
  if (clicks == 1)
    numClicksSpan.innerHTML = 'once';
  else
    numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
  clicks++;
  var numClicksSpan = document.getElementById('numClicks');
  if (clicks == 1)
    numClicksSpan.innerHTML = 'once';
  else
    numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
  clicks++;
  var numClicksSpan = document.getElementById('numClicks');
  if (clicks == 1)
    numClicksSpan.innerHTML = 'once';
  else
    numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```



```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
  clicks++;
  var numClicksSpan = document.getElementById('numClicks');
  if (clicks == 1)
    numClicksSpan.innerHTML = 'once';
  else
    numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

# Event-Driven Programming

---

- Ordinarily we think of a program as a sequence of instructions and function calls
- **Event-Driven programming** is when a program's behavior is based on *events*
- In web programming, these events are generally user actions
- Different events/actions invoke different **callback functions** which handle that event/action

# Event-Driven Programming

---

- Event-Driven programming is a form of asynchronous programming

# Event-Driven Programming

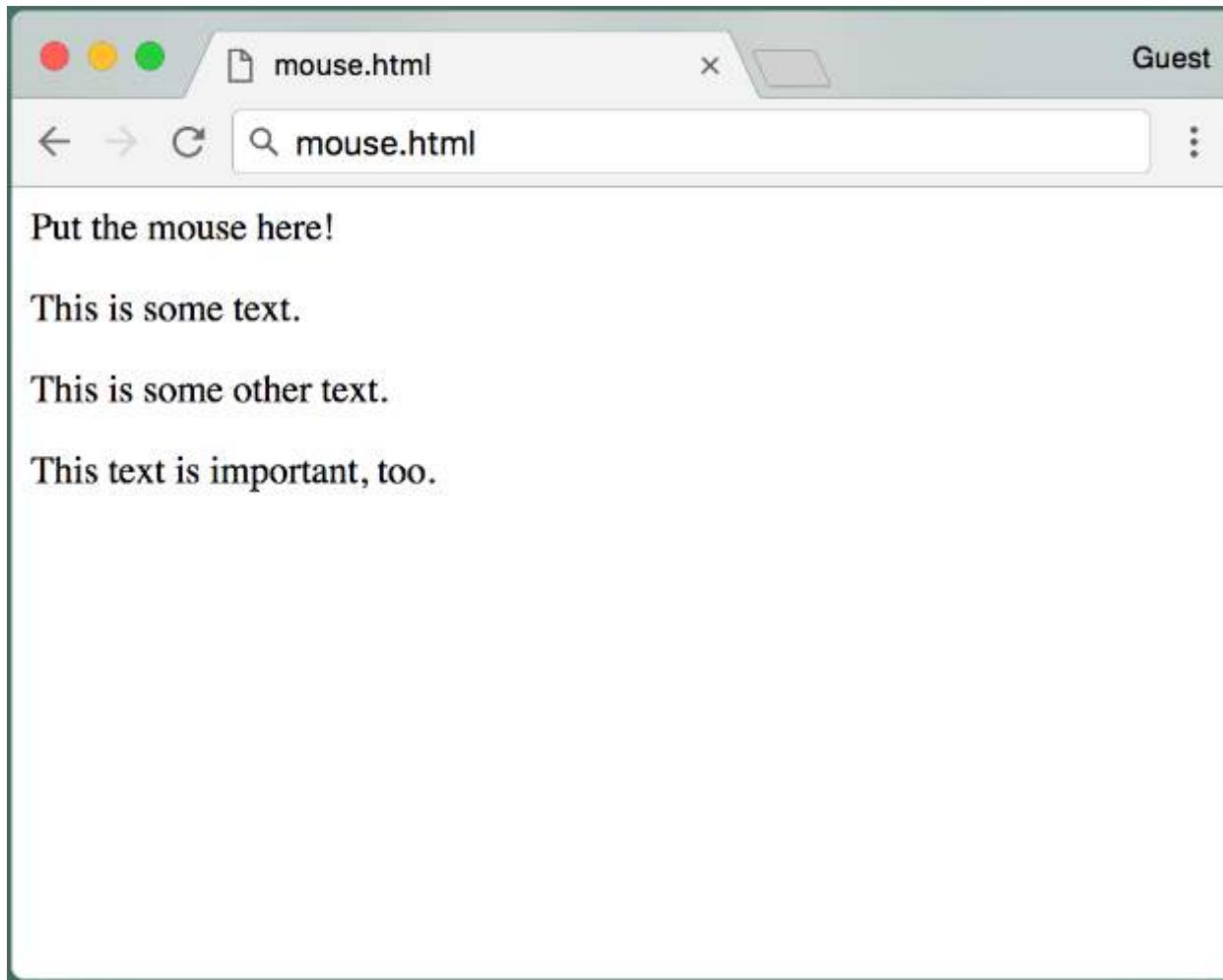
---

- Event-Driven programming is a form of asynchronous programming
  - Synchronous Way:
    - You are expecting a user input
    - You continuously re-check a text field until the user has put in the required information
    - You run some code on the user input

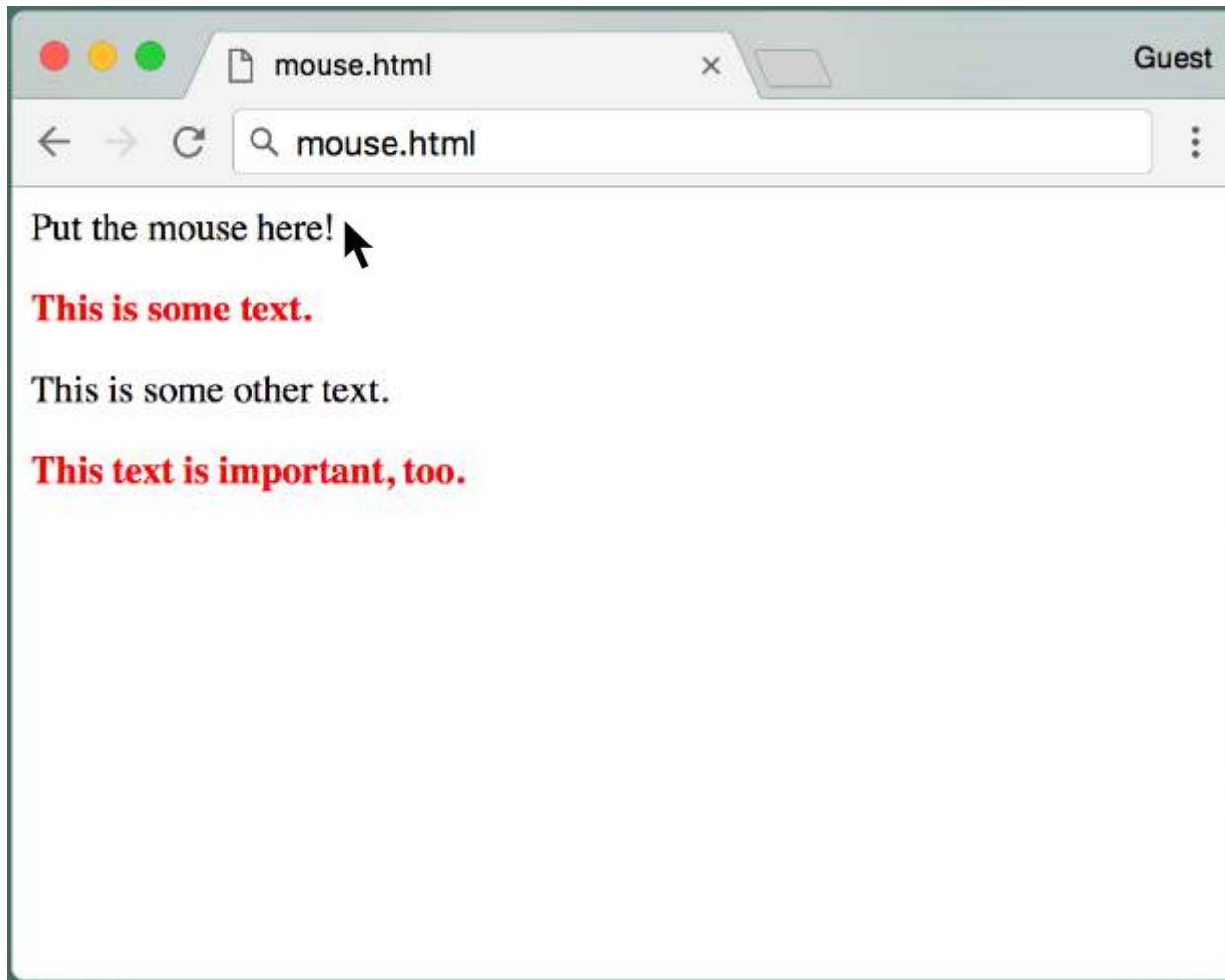
# Event-Driven Programming

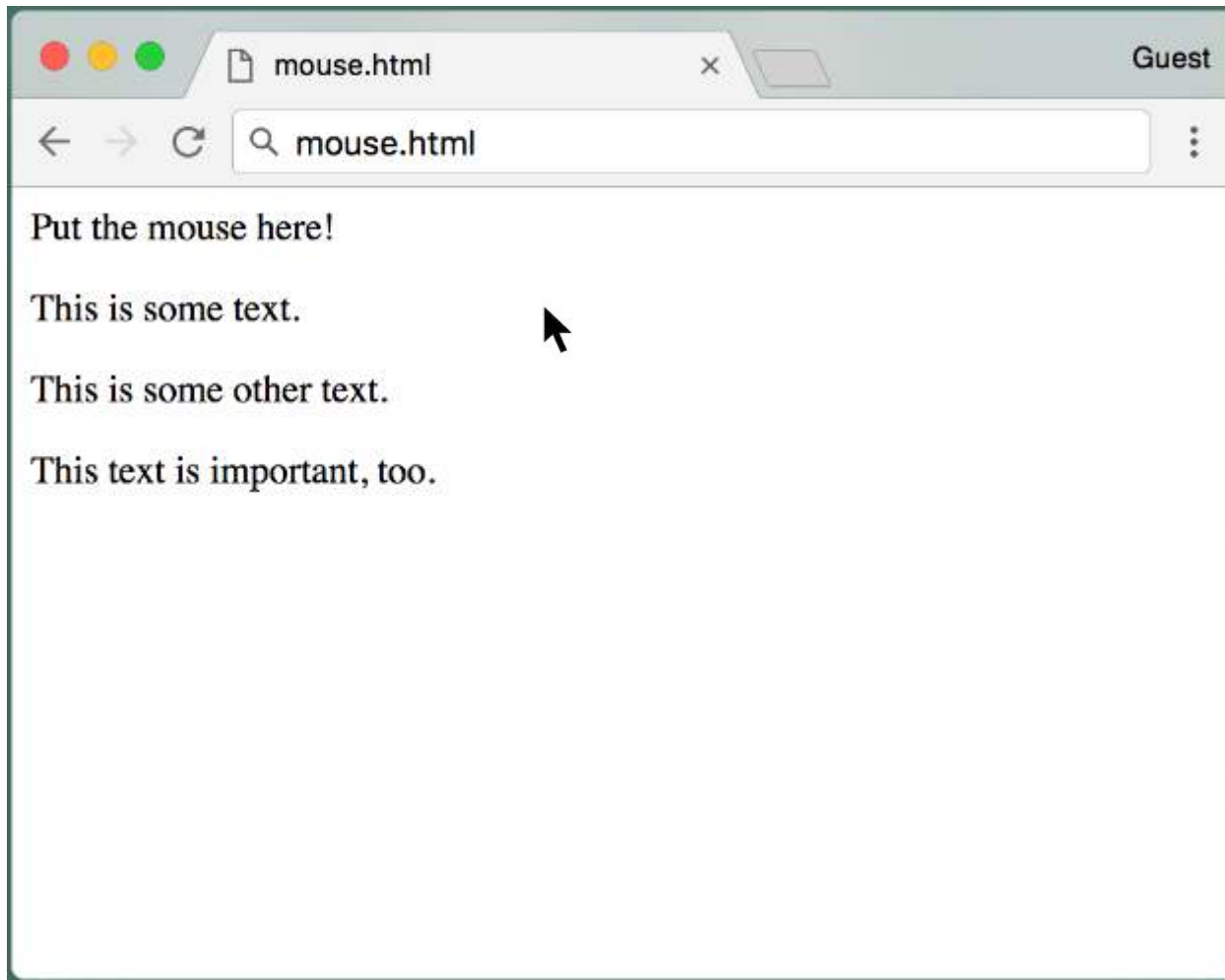
---

- Event-Driven programming is a form of asynchronous programming
  - Synchronous Way:
    - You are expecting a user input
    - You continuously re-check a text field until the user has put in the required information
    - You run some code on the user input
  - Event-Driven/Asynchronous Way:
    - You are expecting a user input
    - You tell your browser to let your program know when the user has put in the required information
    - You (possibly) run other code until your browser notifies you
    - When the user has entered the information, you run the associated callback function









```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>  
<div class="highlightText"> This is some text. </div> <p>  
<div> This is some other text. </div> <p>  
<div class="highlightText"> This text is important, too.</div>  
  
<script>  
  function makeBold() {  
    var divs = document.getElementsByClassName('highlightText');  
    for (var i = 0; i < divs.length; i++) {  
      divs[i].style.color = 'red';  
      divs[i].style.fontWeight = 'bold';  
    }  
  }  
  
  function restore() {  
    var divs = document.getElementsByClassName('highlightText');  
    for (var i = 0; i < divs.length; i++) {  
      divs[i].style.color = 'black';  
      divs[i].style.fontWeight = 'normal';  
    }  
  }  
  
  var mouseOverMeDiv = document.getElementById('mouseOverMe');  
  mouseOverMeDiv.addEventListener('mouseover', makeBold);  
  mouseOverMeDiv.addEventListener('mouseout', restore);  
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>
```

```
<script>
function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
        divs[i].style.color = 'red';
        divs[i].style.fontWeight = 'bold';
    }
}

function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
        divs[i].style.color = 'black';
        divs[i].style.fontWeight = 'normal';
    }
}

var mouseOverMeDiv = document.getElementById('mouseOverMe');
mouseOverMeDiv.addEventListener('mouseover', makeBold);
mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```



```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```



```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

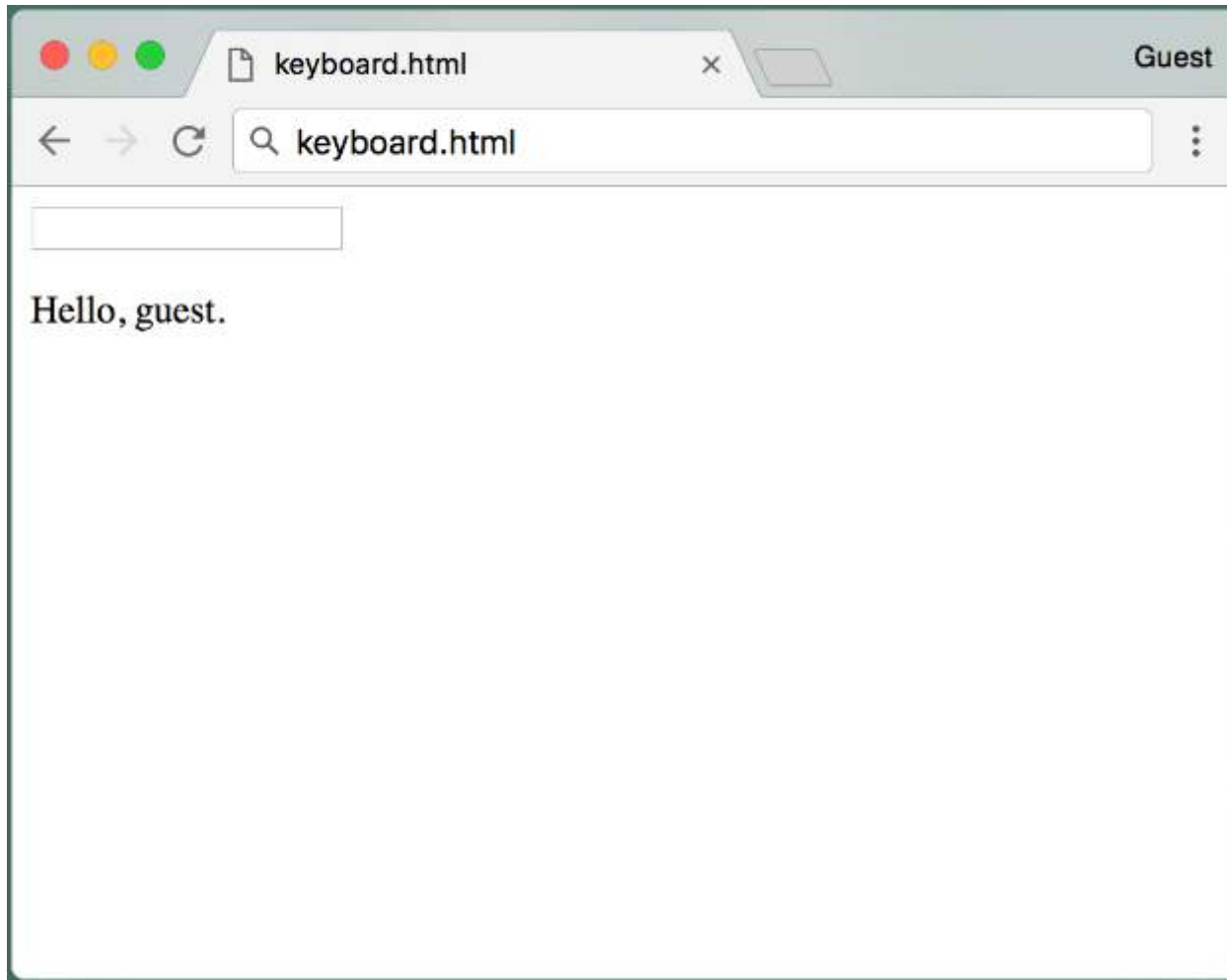
  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```

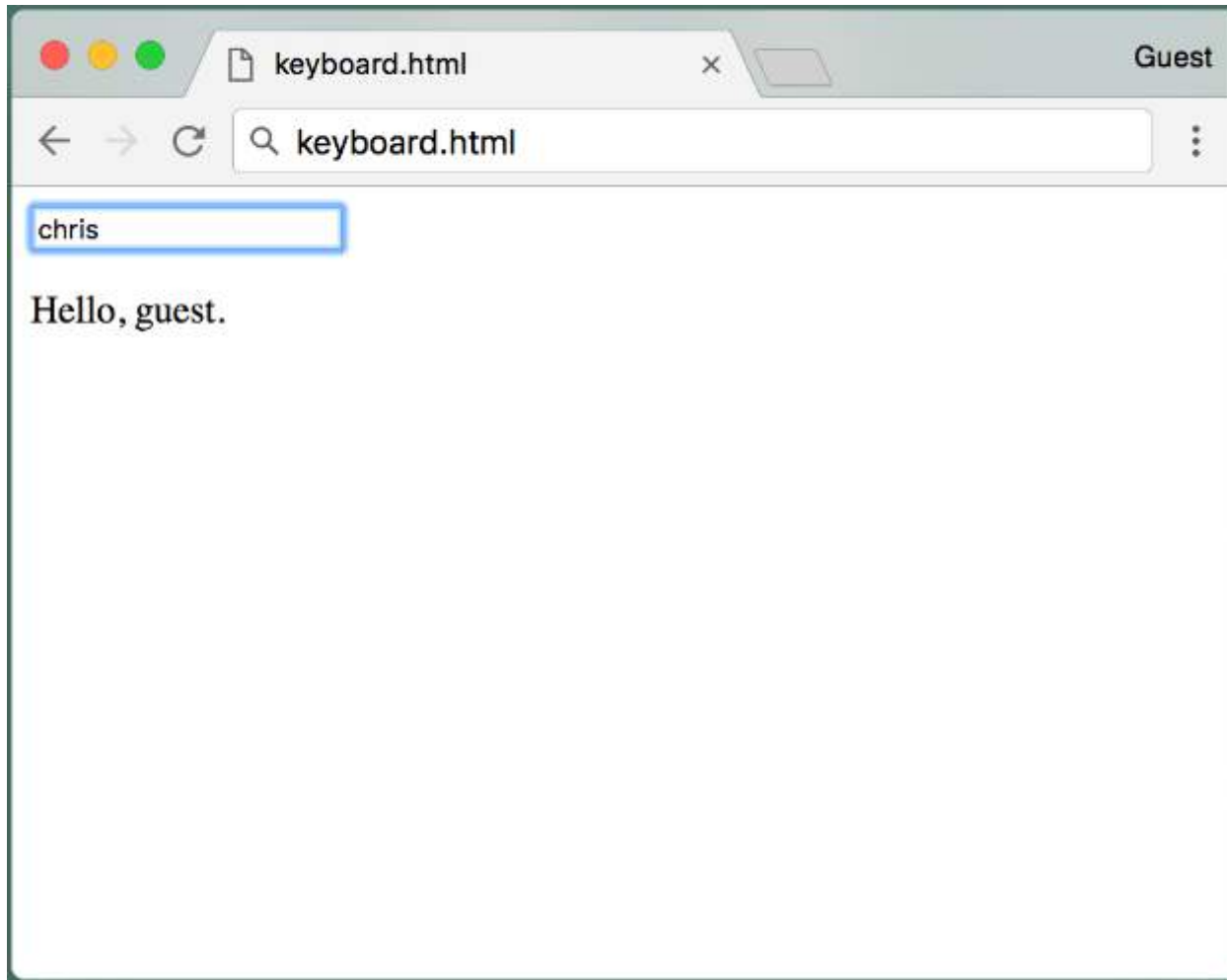
```
<div id="mouseOverMe"> Put the mouse here! </div> <p>
<div class="highlightText"> This is some text. </div> <p>
<div> This is some other text. </div> <p>
<div class="highlightText"> This text is important, too.</div>

<script>
  function makeBold() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'red';
      divs[i].style.fontWeight = 'bold';
    }
  }

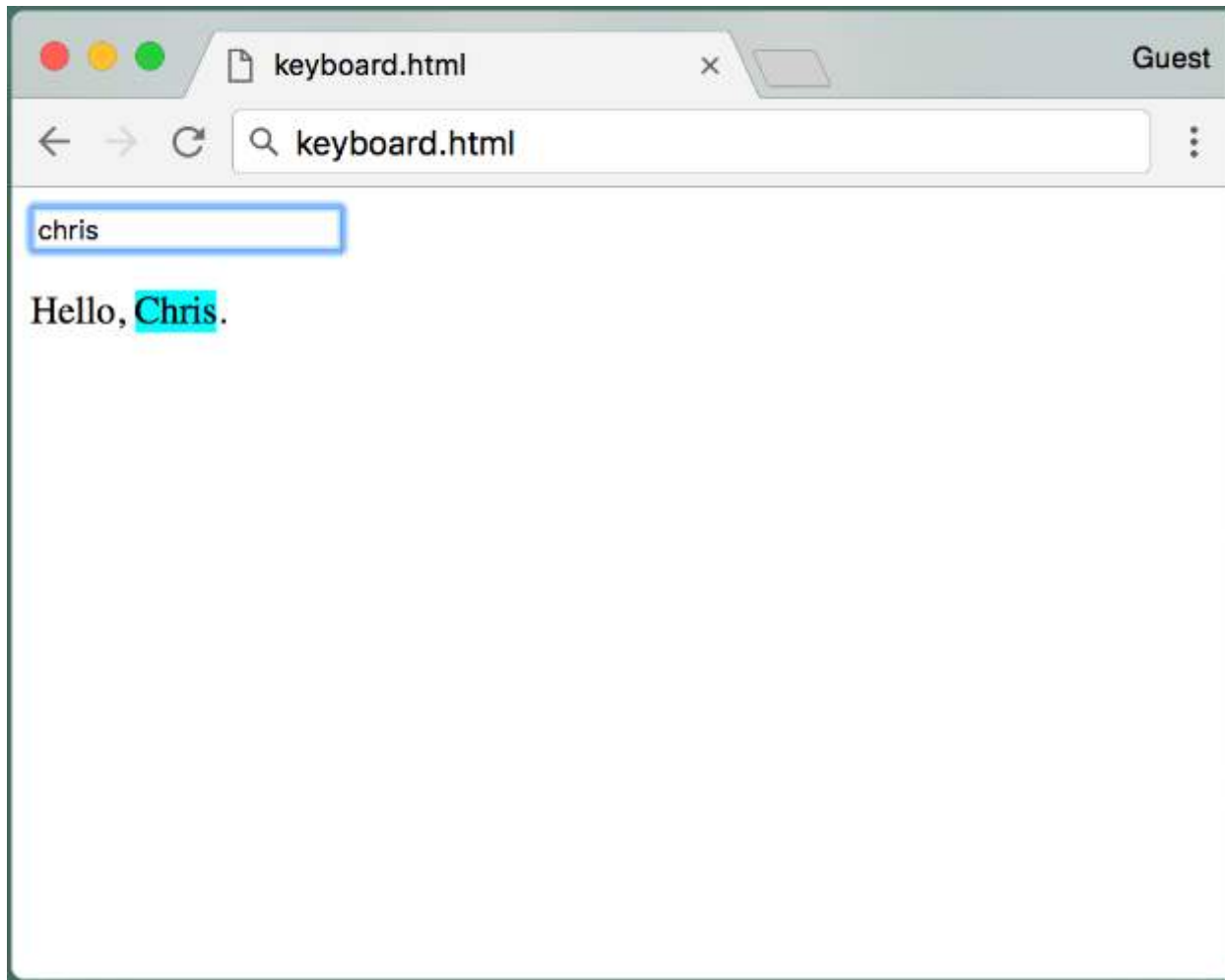
  function restore() {
    var divs = document.getElementsByClassName('highlightText');
    for (var i = 0; i < divs.length; i++) {
      divs[i].style.color = 'black';
      divs[i].style.fontWeight = 'normal';
    }
  }

  var mouseOverMeDiv = document.getElementById('mouseOverMe');
  mouseOverMeDiv.addEventListener('mouseover', makeBold);
  mouseOverMeDiv.addEventListener('mouseout', restore);
</script>
```









```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) { // 13 is the Enter key
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```



```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```



```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

```
<html>
<body>

<input id="nameInput"></input>
<p>
Hello, <span id="nameField">guest</span>.

<script>

function nameHandler(e) {
  if (e.keyCode == 13) {
    var nameInput = document.getElementById('nameInput');
    var nameField = document.getElementById('nameField');
    nameField.innerHTML = nameInput.value;
    nameField.style.backgroundColor = 'cyan';
    nameField.style.textTransform = 'capitalize';
  }
}

document.addEventListener('keyup', nameHandler);

</script>

</body>
</html>
```

# Summary

---

- We can use **event-driven programming** in JavaScript to modify HTML based on user activity
- We do this by defining **callback functions** and associating them with various events by adding event listeners
  - *element.addEventListener(event, function)*
  - Events: 'click', 'mouseover', 'mouseout', 'keyup'



# Video 2.9

## Chris Murphy



# Review

---

- Previously we've seen how to use JavaScript, the DOM, and event-driven programming to modify HTML based on user activity
- However...
  - different browsers may work in different manners
  - the syntax can be a bit clunky
  - many features are hard to implement
- Is there an easier way?

# jQuery

---

- Simplifies JavaScript usage on webapps
- More intuitive way of DOM manipulation
- Great cross-browser support (Except IE6)
- Additional Utilities
- Effects and Animations
- Customizable plugins

# Using jQuery

---

- Download the latest version of jQuery from [jquery.com](http://jquery.com)
- Add the downloaded .js file to your HTML webpage using a script tag
  - `<script src="jQueryFile.js"></script>`

# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax

# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax
  - `$("*")` **selects all elements**

# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax
  - `$( "* " )` selects all elements
  - **`$( this )` selects the current element**

# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax
  - `$("*")` selects all elements
  - `$(this)` selects the current element
  - **`$("div")` selects all `<div>` elements**

# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax
  - `$("*")` selects all elements
  - `$(this)` selects the current element
  - `$("div")` selects all `<div>` elements
  - **`$(".title")` selects all elements with `class="title"`**



# Selecting DOM Elements

---

- In jQuery, `$` is used to select DOM elements for manipulation, along with basic CSS element syntax
  - `$("*")` selects all elements
  - `$(this)` selects the current element
  - `$("div")` selects all `<div>` elements
  - `$(".title")` selects all elements with `class="title"`
  - **`$("#name")` selects the element with `id="name"`**

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");  
$("#name").append(" World!");
```



# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");  
$("#name").append(" World!");  
$("#name").addClass("greeting");
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");  
$("#name").append(" World!");  
$("#name").addClass("greeting");  
$("#name").hide();
```

# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");  
$("#name").append(" World!");  
$("#name").addClass("greeting");  
$("#name").hide();  
$("#name").show();
```

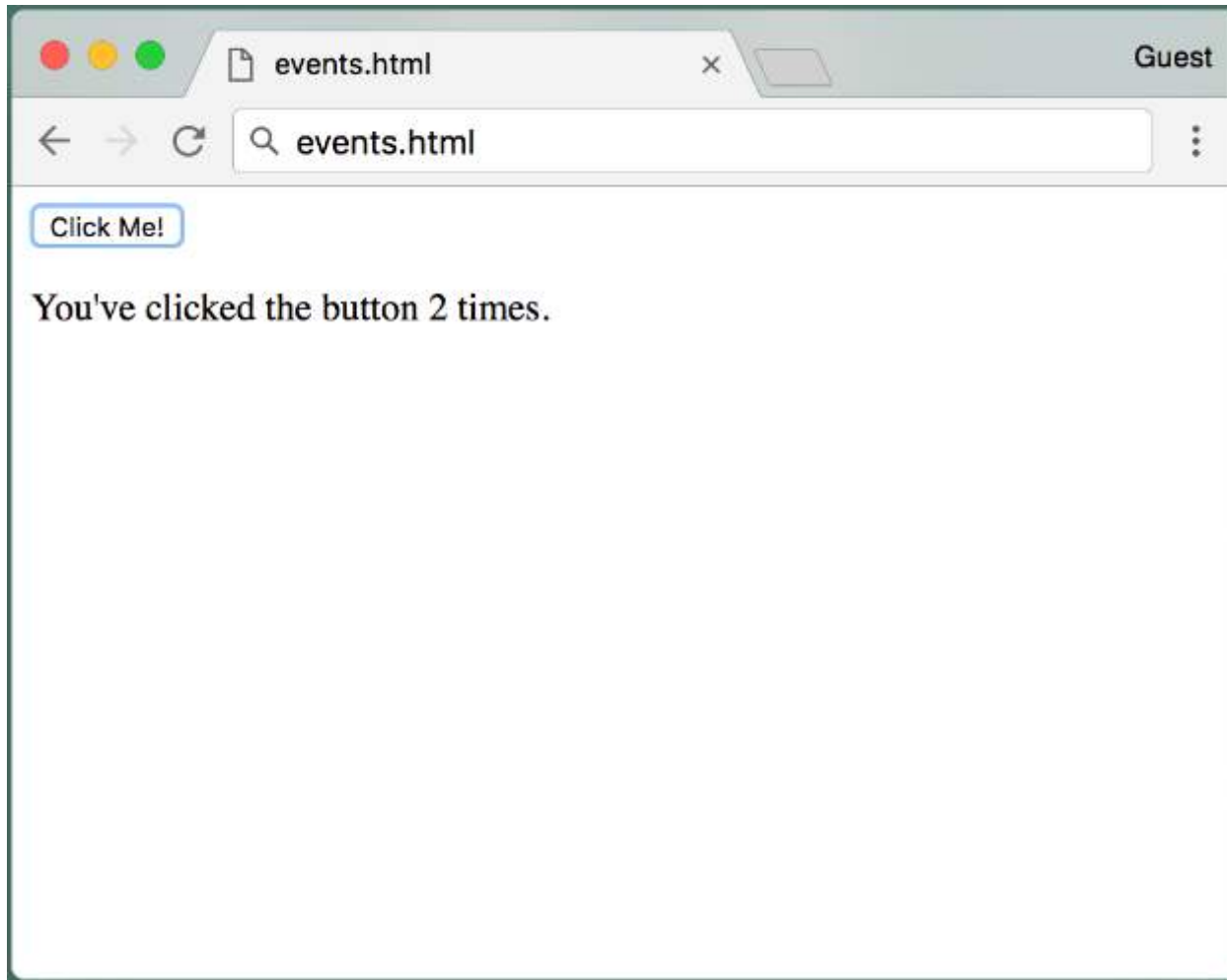
# jQuery DOM Manipulation

---

- To manipulate DOM contents, the general format is `$(selector).action(arguments...)`

```
$("#name").html("Hello");  
$("#name").append(" World!");  
$("#name").addClass("greeting");  
$("#name").hide();  
$("#name").show();
```

- To add an event listener to an element, the general format is `$(selector).event(callback)`



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = document.getElementById('numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.innerHTML = 'once';
    else
        numClicksSpan.innerHTML = clicks + ' times';
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = document.getElementById('clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.addEventListener('click', clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.click(clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.click(clickHandler);

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

var button = $('#clickMe');
button.click(clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

<script>
var clicks = 0;

function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

$('#clickMe').click(clickHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<button id="clickMe">Click Me!</button>
<p>
You've clicked the button <span id="numClicks">0 times</span>.

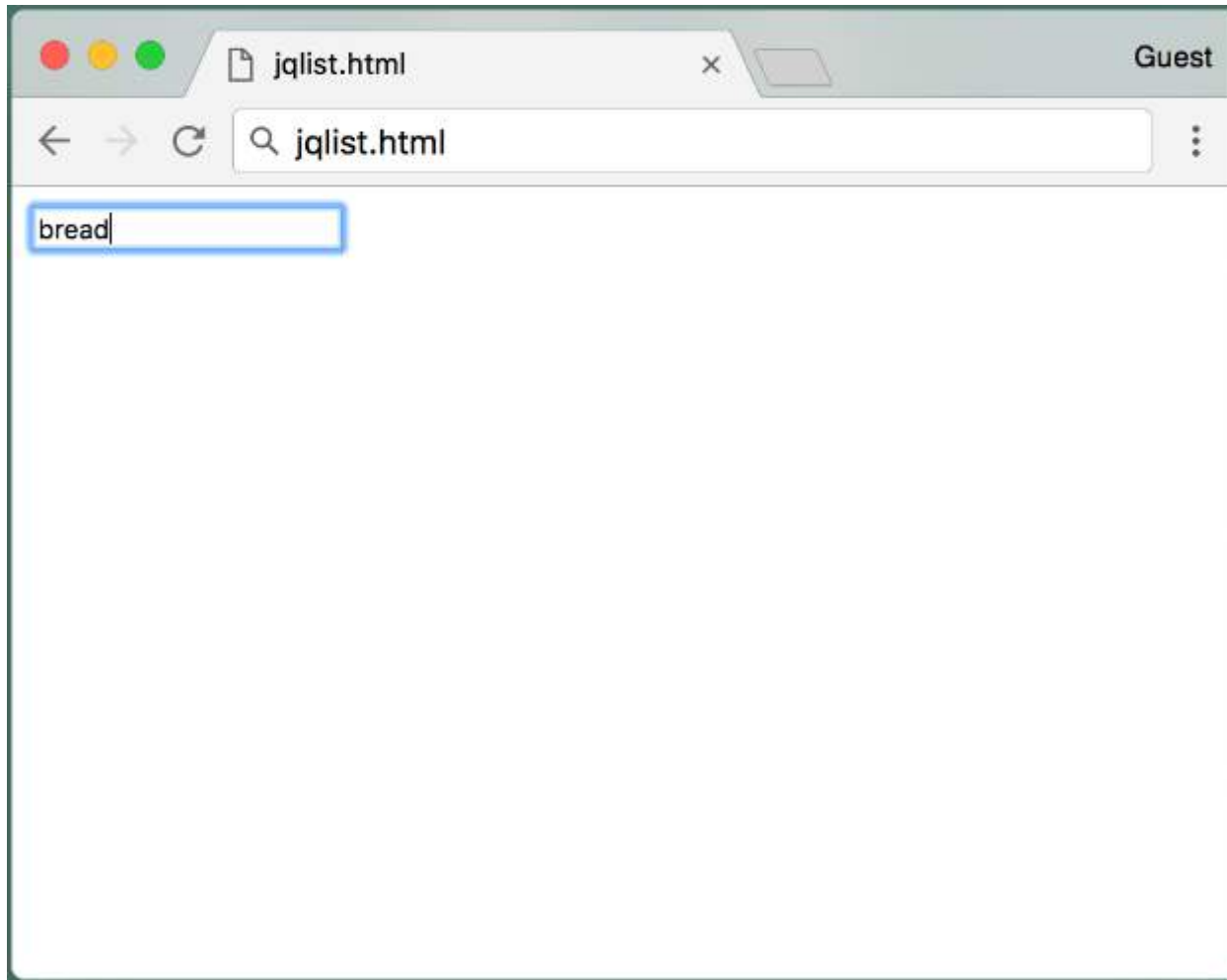
<script>
var clicks = 0;

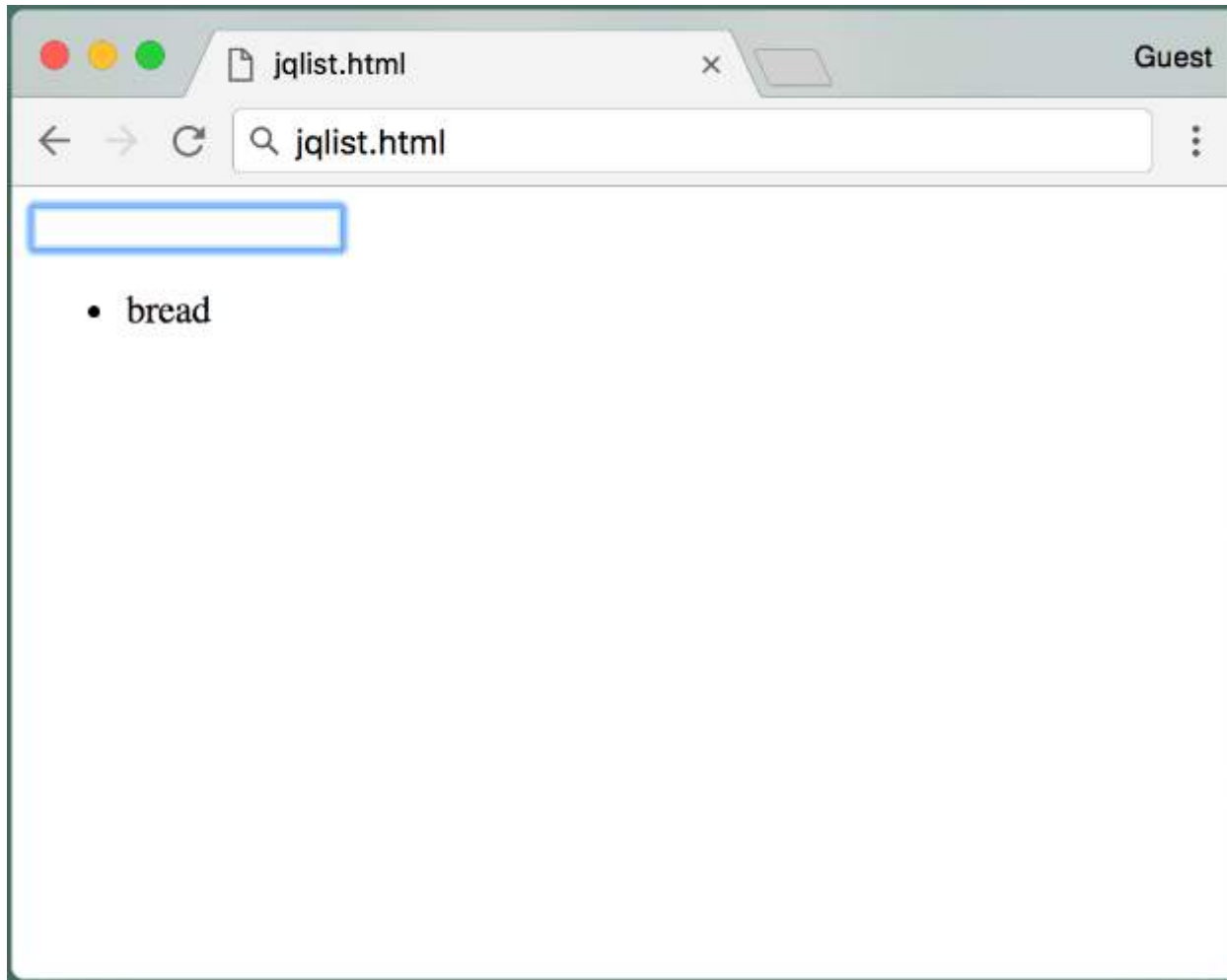
function clickHandler() {
    clicks++;
    var numClicksSpan = $('#numClicks');
    if (clicks == 1)
        numClicksSpan.html('once');
    else
        numClicksSpan.html(clicks + ' times');
}

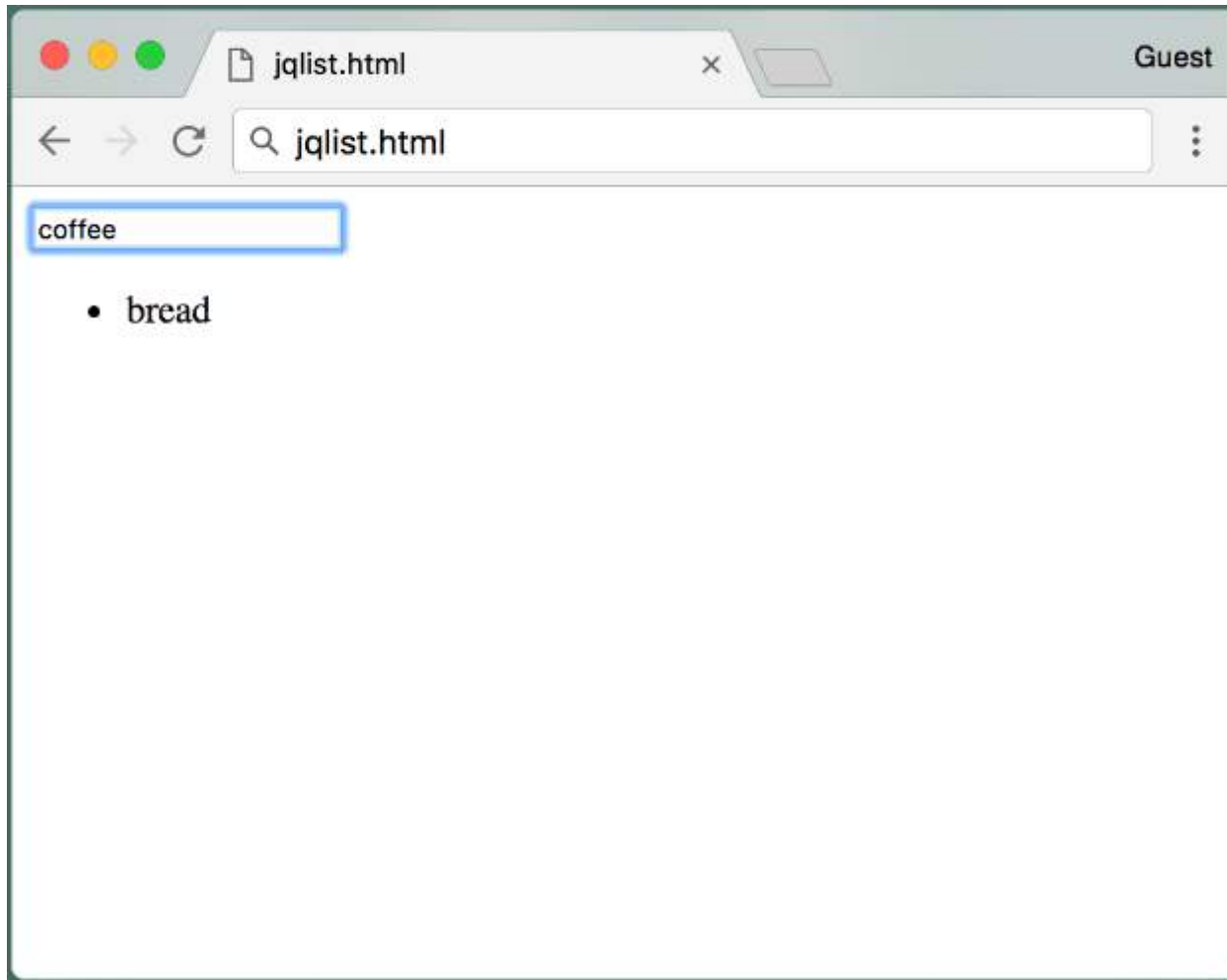
$('#clickMe').click(clickHandler);

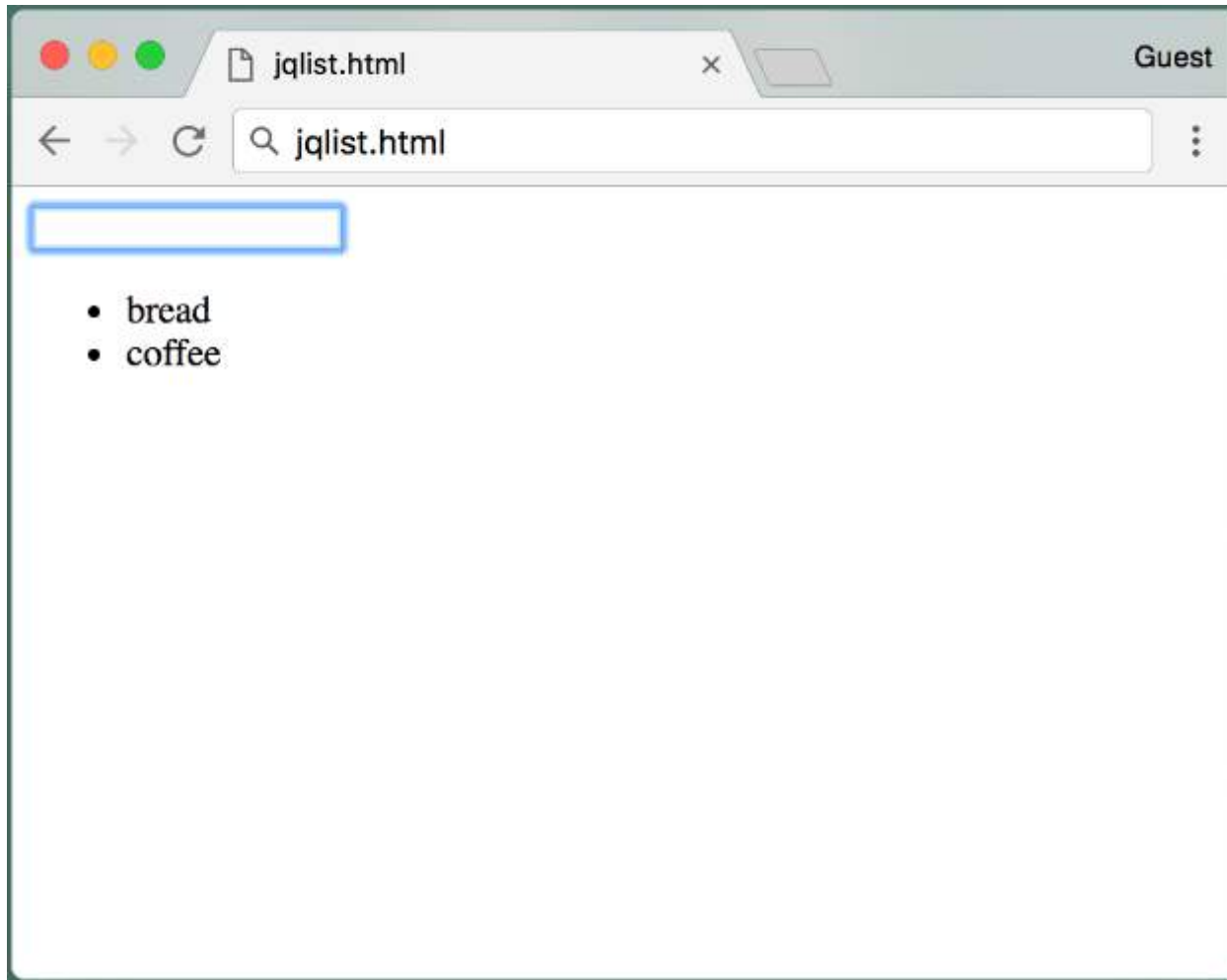
</script>

</body>
</html>
```









```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler (e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
    if (e.keyCode == 13) {
        $('#list').append('<li>' + $('#itemField').val() + '</li>');
        $('#itemField').val('');
    }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

$('#itemField').keyup(keyPressHandler);

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<input id="itemField"></input>
<p>

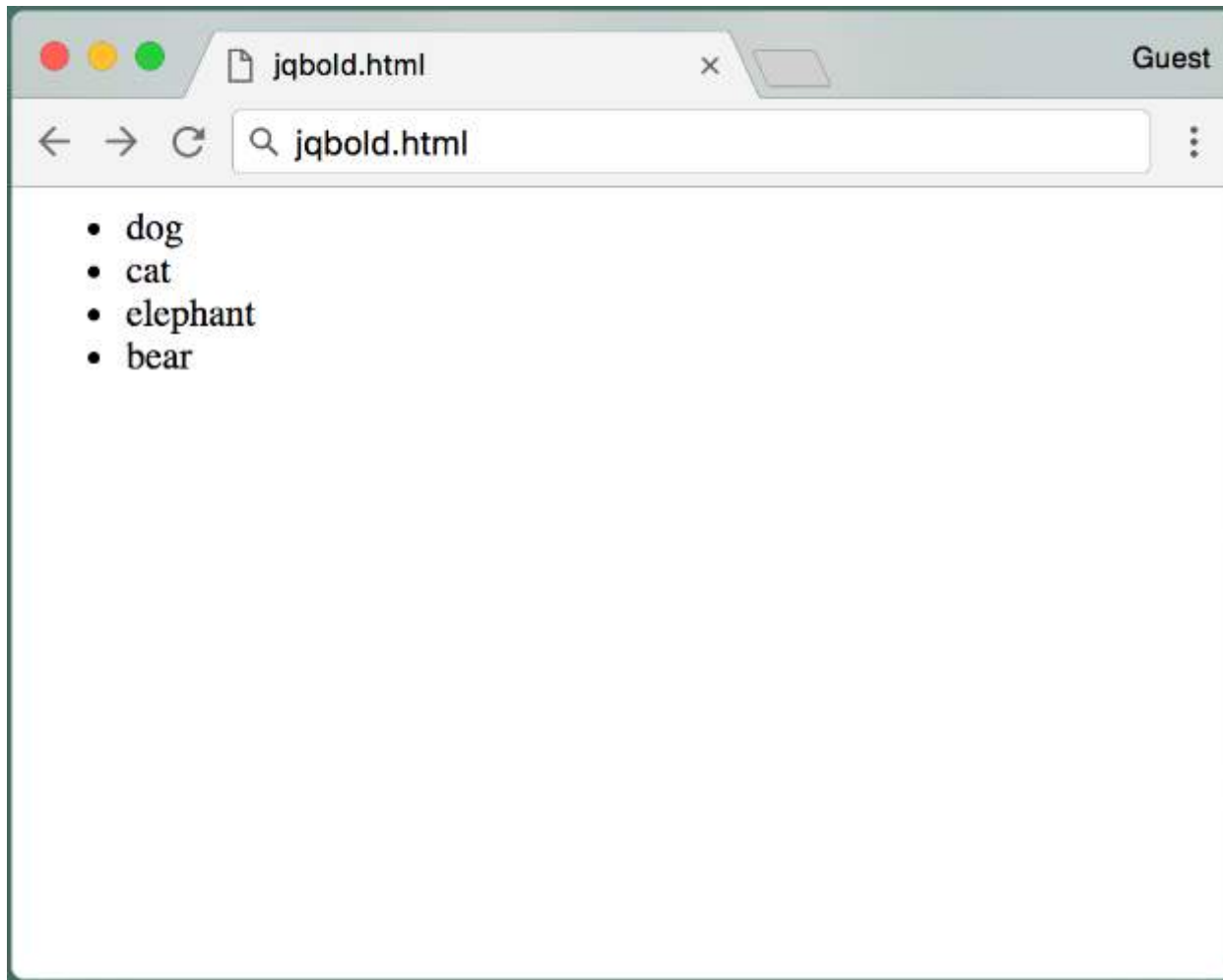
<ul>
<span id="list"></span>
</ul>

<script>
function keyPressHandler(e) {
  if (e.keyCode == 13) {
    $('#list').append('<li>' + $('#itemField').val() + '</li>');
    $('#itemField').val('');
  }
}

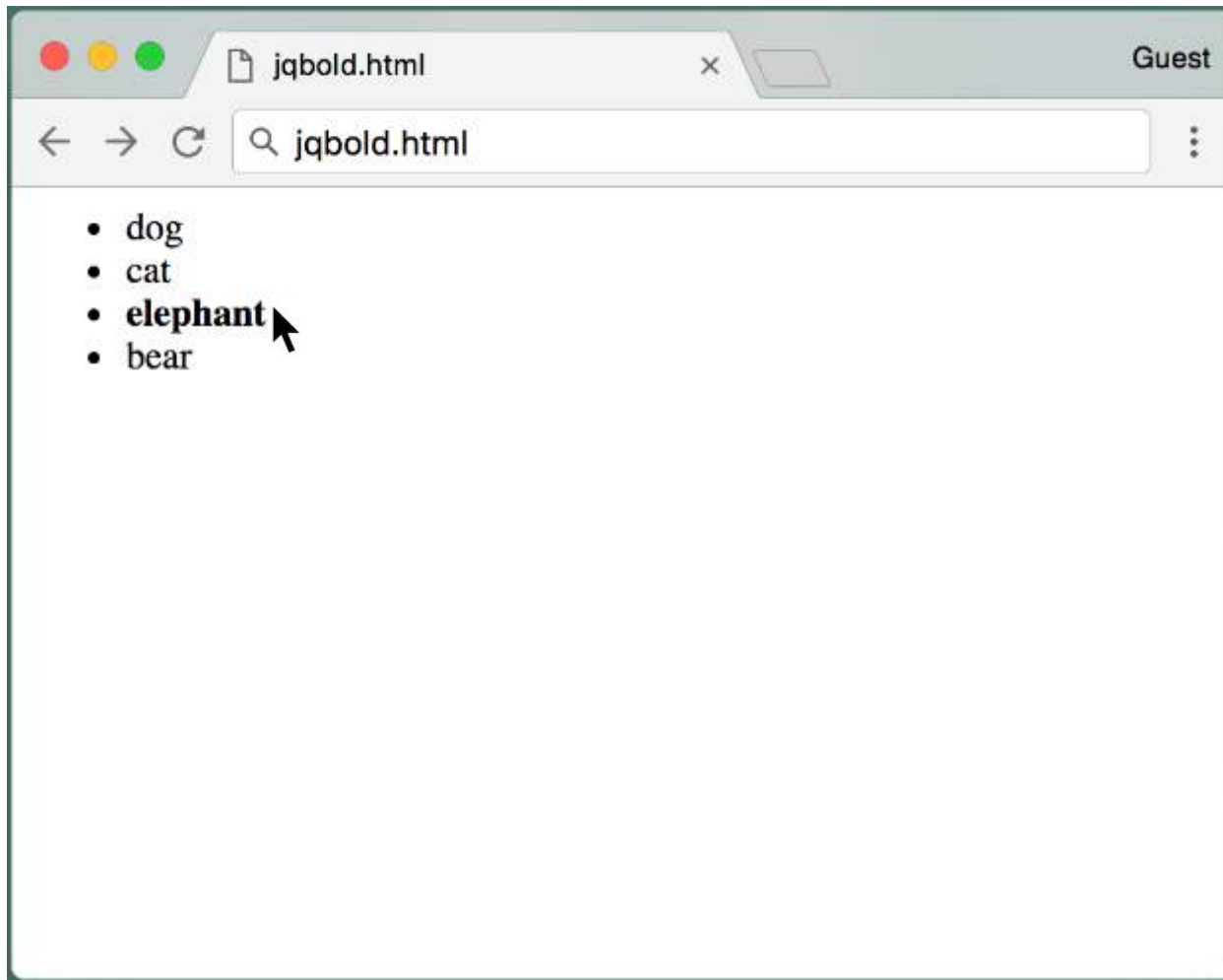
<b>$('#itemField').keyup(keyPressHandler);</b>

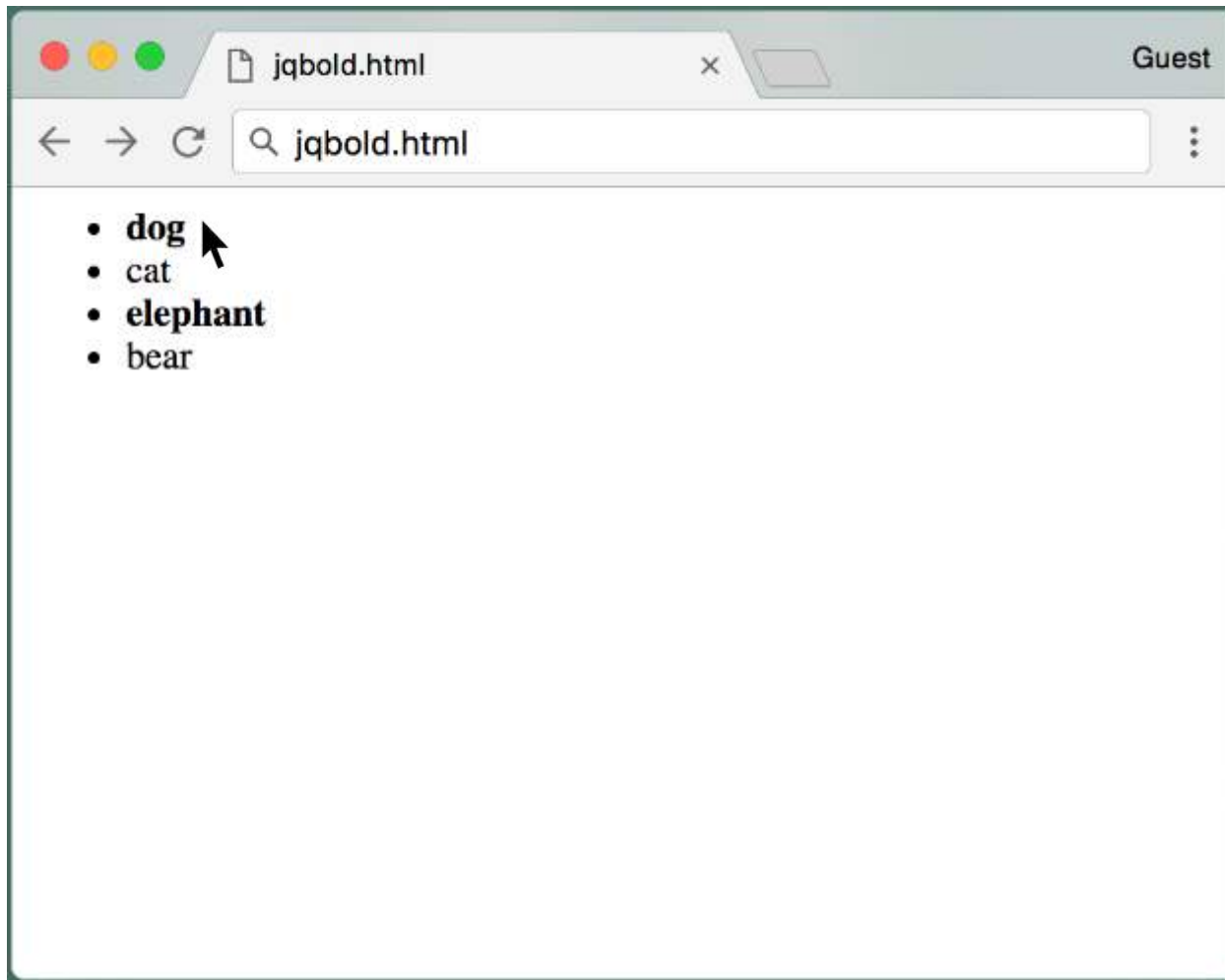
</script>

</body>
</html>
```









```
<html>

<head><script src="jquery.js"></script></head>

<body>

<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').click(function() {
    $(this).css('font-weight', 'bold');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>
<script>
$('li').click(function() {
    $(this).css('font-weight', 'bold');
});
</script>
</body>
</html>
```

```
<html>

<head><script src="jquery.js"></script></head>

<body>

<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').click(function() {
    $(this).css('font-weight', 'bold');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>
<script>
$('li').click(function() {
    $(this).css('font-weight', 'bold');
});
</script>
</body>
</html>
```

```
<html>

<head><script src="jquery.js"></script></head>

<body>

<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').click(function() {
    $(this).css('font-weight', 'bold');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>
<script>
$('li').click(function() {
    $(this).css('font-weight', 'bold');
});
</script>
</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>
<script>
$('li').click(function () {
    $(this).css('font-weight', 'bold');
});
</script>
</body>
</html>
```

```
<html>

<head><script src="jquery.js"></script></head>

<body>

<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').click(function() {
    $(this).css('font-weight', 'bold');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>
<script>
$('li').click(function() {
    $(this).css('font-weight', 'bold');
});
</script>
</body>
</html>
```

```
<html>

<head><script src="jquery.js"></script></head>

<body>

<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').click(function() {
    $(this).css('font-weight', 'bold');
});

</script>

</body>
</html>
```

# Summary

---

- jQuery is a powerful library that allows us to select DOM elements using CSS notation
- We can then modify their content and appearance programmatically
- We can also register event listeners for different elements



# Video 2.10

## Chris Murphy

# Review

---

- Previously we saw how to use jQuery to select and modify DOM elements and add event listeners
- How else can we specify selectors and add event listeners?

- dog
- cat
- elephant
- bear



- dog
- cat
- elephant
- bear

- dog
- cat
- elephant
- **bear**

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$( 'li' ).mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$( 'li' ).mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').mouseenter(function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
});

$('li').mouseleave(function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
});

</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').hover(
  function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
  },
  function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
  });
</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>

$('li').hover(
  function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
  },
  function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
  });
</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>
$('li').on({
  mouseenter: function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
  },
  mouseleave: function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
  },
  click: function() {
    $(this).css('background-color', 'yellow');
  }
});
</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>
$('li').on({
  mouseenter: function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
  },
  mouseleave: function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
  },
  click: function() {
    $(this).css('background-color', 'yellow');
  }
});
</script>

</body>
</html>
```

```
<html>
<head><script src="jquery.js"></script></head>
<body>
<ul>
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<script>
$('li').on({
  mouseenter: function() {
    $(this).css('color', 'red');
    $(this).css('font-size', '120%');
  },
  mouseleave: function() {
    $(this).css('color', 'black');
    $(this).css('font-size', '100%');
  },
  click: function() {
    $(this).css('background-color', 'yellow');
  }
});
</script>

</body>
</html>
```

- dog
- cat
- elephant
- bear
  
- canary
- eagle

```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul>
<li class="highlight">dog</li>
<li class="highlight">cat</li>
<li class="highlight">elephant</li>
<li class="highlight">bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("li.highlight").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul>
<li class="highlight">dog</li>
<li class="highlight">cat</li>
<li class="highlight">elephant</li>
<li class="highlight">bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("li.highlight").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul>
<li class="highlight">dog</li>
<li class="highlight">cat</li>
<li class="highlight">elephant</li>
<li class="highlight">bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("li.highlight").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul>
<li class="highlight">dog</li>
<li class="highlight">cat</li>
<li class="highlight">elephant</li>
<li class="highlight">bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("#li.highlight").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```





```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



```
<html>
<head><script src="jquery.js"></script></head>

<body>
<ul class="highlight">
<li>dog</li>
<li>cat</li>
<li>elephant</li>
<li>bear</li>
</ul>

<ul>
<li>canary</li>
<li>eagle</li>
</ul>

<script>

$("ul.highlight").find("li").on({
mouseenter: function() { . . . },
mouseleave: function() { . . . },
click: function() { . . . }
});

</script>

</body>
</html>
```



# Advanced Selectors

---

- `$ (someNodes) .find (selector)` will search *someNodes*' children for selector.

# Advanced Selectors

---

- `$(someNodes).find(selector)` will search *someNodes*' children for selector.
- `$` selectors can be chained
  - `$("#div.book")` selects the div with class="book"
  - `$("#div, .book")` selects all divs **and** all elements with class="book"

# Advanced Selectors

---

- `$(someNodes).find(selector)` will search *someNodes*' children for selector.
- `$` selectors can be chained
  - `$("#div.book")` selects the div with class="book"
  - `$("#div, .book")` selects all divs **and** all elements with class="book"
- `:` can be used to specify element properties
  - `$("#p:hidden")` selects all `<p>` elements that are visually hidden



# Advanced Selectors

---

- `$(someNodes).find(selector)` will search *someNodes*' children for selector.
- `$` selectors can be chained
  - `$("#div.book")` selects the div with class="book"
  - `$("#div, .book")` selects all divs **and** all elements with class="book"
- `:` can be used to specify element properties
  - `$("#p:hidden")` selects all `<p>` elements that are visually hidden
- These selectors are all CSS selectors!
  - All other CSS selectors also work in `$`

# Summary

---

- jQuery can be used to retrieve and modify DOM elements and add event listeners
  
- We can use advanced selectors and handlers to simplify functionality



# Video 2.11

## Chris Murphy

# Review

---

- Previously we saw how to use jQuery to retrieve and modify DOM elements and add event listeners
- What other types of events can we handle?

# User Events in the Browser

---

Event Type	Events
Mouse	click, dblclick, mousedown, mouseup, mouseover, mouseout

# User Events in the Browser

---

Event Type	Events
Mouse	click, dblclick, mousedown, mouseup, mouseover, mouseout
Keyboard	keydown, keypress, keyup

# User Events in the Browser

---

Event Type	Events
Mouse	click, dblclick, mousedown, mouseup, mouseover, mouseout
Keyboard	keydown, keypress, keyup
Form	focus, blur, change, reset, submit

# User Events in the Browser

---

Event Type	Events
Mouse	click, dblclick, mousedown, mouseup, mouseover, mouseout
Keyboard	keydown, keypress, keyup
Form	focus, blur, change, reset, submit
Window / Element	load, resize, scroll, unload



▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new animal.

▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new bird.

Male ▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new cat.

Male  
 Female

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new cat.

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new female cat.

Female ▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new happy female cat.

Female ▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new happy, smart female cat.

Female ▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new happy, smart, cute female cat.



Female ▾

Dog  Cat  Bird

Happy  Cute  Smart

I'd like to buy a new happy, cute female cat.

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<b><input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
</b>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```



```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<head>
<script src="jquery.js"></script>
</head>

<body>

<form>

<select name="choose">
  <option value="male">Male</option>
  <option value="female">Female</option>
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>
<p>
<input type="checkbox" value="happy">Happy</input>
<input type="checkbox" value="cute">Cute</input>
<input type="checkbox" value="smart">Smart</input>

</form>

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

```
<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.
```

### <script>

```
// handling select box
$("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
});

// handling radio buttons
$("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
        $('#speciesSpan').html($(this).val());
    }
});
```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $ ("select[name='choose']") .change(function () {
    $ ('#genderSpan') .html ($ (this) .val ());
  });

  // handling radio buttons
  $ ("input:radio[name='species']") .change(function () {
    if ($ (this) .prop ('checked')) {
      $ ('#speciesSpan') .html ($ (this) .val ());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```



```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("#input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```



```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```



```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```

<select name="choose">
  . . .
</select>
<p>
<input type="radio" name="species" value="dog">Dog</input>
<input type="radio" name="species" value="cat">Cat</input>
<input type="radio" name="species" value="bird">Bird</input>

. . .

<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>

  // handling select box
  $("select[name='choose']").change(function() {
    $('#genderSpan').html($(this).val());
  });

  // handling radio buttons
  $("input:radio[name='species']").change(function() {
    if ($(this).prop('checked')) {
      $('#speciesSpan').html($(this).val());
    }
  });

```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$( 'input:checkbox' ).change( function() {
    var value = $(this).val();
    if ( $(this).prop('checked') ) {
        allChecked.push(value);
    }
    else {
        var index = allChecked.indexOf(value);
        if (index != -1)
            allChecked.splice(index, 1);
    }
    $( '#featureSpan' ).html('');
    for (var i = 0; i < allChecked.length; i++) {
        $( '#featureSpan' ).append(allChecked[i]);
        if (i < allChecked.length - 1)
            $( '#featureSpan' ).append(', ');
        else
            $( '#featureSpan' ).append(' ');
    }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```



```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
    var value = $(this).val();
    if ($(this).prop('checked')) {
        allChecked.push(value);
    }
    else {
        var index = allChecked.indexOf(value);
        if (index != -1)
            allChecked.splice(index, 1);
    }
    $('#featureSpan').html('');
    for (var i = 0; i < allChecked.length; i++) {
        $('#featureSpan').append(allChecked[i]);
        if (i < allChecked.length - 1)
            $('#featureSpan').append(', ');
        else
            $('#featureSpan').append(' ');
    }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else { // this element was unchecked
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
  . . .
  var allChecked = [];

  $('input:checkbox').change(function() {
    var value = $(this).val();
    if ($(this).prop('checked')) {
      allChecked.push(value);
    }
    else {
      var index = allChecked.indexOf(value);
      if (index != -1)
        allChecked.splice(index, 1);
    }
    $('#featureSpan').html('');
    for (var i = 0; i < allChecked.length; i++) {
      $('#featureSpan').append(allChecked[i]);
      if (i < allChecked.length - 1)
        $('#featureSpan').append(', ');
      else
        $('#featureSpan').append(' ');
    }
  });
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```



```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
$('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

```
<p>
I'd like to buy a new <span id="featureSpan"></span>
<span id="genderSpan"></span> <span id="speciesSpan">animal</span>.

<script>
. . .
var allChecked = [];

$('input:checkbox').change(function() {
  var value = $(this).val();
  if ($(this).prop('checked')) {
    allChecked.push(value);
  }
  else {
    var index = allChecked.indexOf(value);
    if (index != -1)
      allChecked.splice(index, 1);
  }
  $('#featureSpan').html('');
  for (var i = 0; i < allChecked.length; i++) {
    $('#featureSpan').append(allChecked[i]);
    if (i < allChecked.length - 1)
      $('#featureSpan').append(', ');
    else
      $('#featureSpan').append(' ');
  }
});
</script>
```

**Validate Password**

Please fix the following errors:

- The password must contain a number
- The password must be at least 10 characters long

Validate Password



Please fix the following errors:

- The password must be at least 10 characters long



The password is okay!

```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```

```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```

```
<head>  
<script src="jquery.js"></script>
```

```
<style>
```

```
.errorText {  
color: red;  
}
```

```
.errorBox {  
border: 2px solid red;  
}
```

```
.goodBox {  
border: 2px solid green;  
}
```

```
</style>
```

```
</head>
```

```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```

```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```

```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```



```
<head>
<script src="jquery.js"></script>

<style>

.errorText {
color: red;
}

.errorBox {
border: 2px solid red;
}

.goodBox {
border: 2px solid green;
}

</style>

</head>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
```

```
<input type="password" name="password"></input>
```

```
<br>
```

```
<span id="errorMessage" class="errorText" hidden>
```

```
  Please fix the following errors:</span>
```

```
<ul>
```

```
<li id="needsNumber" class="errorText" hidden>
```

```
  The password must contain a number</li>
```

```
<li id="atLeast10Chars" class="errorText" hidden>
```

```
  The password must be at least 10 characters long</li>
```

```
</ul>
```

```
<span id="successMessage" hidden>The password is okay!</span>
```

```
<p>
```

```
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```



```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<b><button name="submit">Validate Password</button></b>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>
```

```
$ ("button[name='submit']").click(function() {  
    var passwordField = $("input[name='password']");  
    var password = passwordField.val();  
    var isOkay = true;  
    if (password.length < 10) {  
        isOkay = false;  
        $('#atLeast10Chars').show();  
    }  
    if (/\\d/.test(password) == false) {  
        isOkay = false;  
        $('#needsNumber').show();  
    }  
    if (isOkay == false) {  
        $('#successMessage').hide();  
        $('#errorMessage').show();  
        passwordField.removeClass("goodBox").addClass("errorBox");  
    }  
    else {  
        $('.errorText').hide();  
        $('#successMessage').show();  
        passwordField.removeClass("errorBox").addClass("goodBox");  
    }  
    return false;  
});
```

```
</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```



```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```



```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```



```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<body>
<input type="password" name="password"></input>
<br>
<span id="errorMessage" class="errorText" hidden>
  Please fix the following errors:</span>
<ul>
<li id="needsNumber" class="errorText" hidden>
  The password must contain a number</li>
<li id="atLeast10Chars" class="errorText" hidden>
  The password must be at least 10 characters long</li>
</ul>
<span id="successMessage" hidden>The password is okay!</span>
<p>
<button name="submit">Validate Password</button>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```



```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else { // isOkay == true
        $('#errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```



```
<script>

$("button[name='submit']").click(function() {
    var passwordField = $("input[name='password']");
    var password = passwordField.val();
    var isOkay = true;
    if (password.length < 10) {
        isOkay = false;
        $('#atLeast10Chars').show();
    }
    if (/\\d/.test(password) == false) {
        isOkay = false;
        $('#needsNumber').show();
    }
    if (isOkay == false) {
        $('#successMessage').hide();
        $('#errorMessage').show();
        passwordField.removeClass("goodBox").addClass("errorBox");
    }
    else {
        $('.errorText').hide();
        $('#successMessage').show();
        passwordField.removeClass("errorBox").addClass("goodBox");
    }
    return false;
});

</script>
```

# Summary

---

- jQuery can be used to retrieve and modify DOM elements and add event listeners
- We have seen how to handle events from the mouse, keyboard, and forms

# Review: Week 2

---

- **JavaScript:** a general-purpose, easy-to-use programming language
- **DOM:** representation of structure of HTML page, which can be manipulated using JavaScript
- **jQuery:** library that simplifies accessing/using the DOM