Week 04 Notes:

Chapter 8 Forms

Form controls

- input fields, select menus, and buttons.
- Usually sent to the backend after submitted but can be verified frontend through JS

Accessing form elements

- document.forms that returns an HTML collection of all the forms in the document in the order they appear
- const form = document.forms.search; using name .search to find the specific form
- const form = document.forms[0]; is the same as
- const form = document.getElementsByTagname('form')[0];

Form validation

Form validation is the process of checking whether a user has entered the information into a form correctly. Examples of the types of validation that occur include ensuring that:

- A required field is completed
- An email address is valid
- A number is entered when numerical data is required
- A password is at least a minimum number of characters

Chapter Summary

- Forms are the primary method used for entering data into a browser.
- Forms have a variety of controls that are used for entering different types of information.
- HTML5 has a large number of new input types that are beginning to be implemented in modern browsers.

- document.forms will return an HTML collection of all the forms on a page.
- form.elements will return an HTML collection of all the elements contained within a form.
- Forms have focus, blur, and change events that fire as a user interacts with the form.
- Forms also have a submit event that can be used to intercept a form before it's been submitted.
- The information entered into a form can be read or updated using the value property of the form controls.
- The HTML5 form validation API can be used to automatically validate a form, but only at a basic level, so a custom validation script may be required.
- In the next chapter, we'll be taking a look at the window object.

Chapter 12

Encapsulation

- The concept of encapsulation: the inner workings are kept hidden inside the object and only the essential functionalities are exposed to the end user,

Polymorphism

- The concept of polymorphism: the same process can be used for different objects. In OOP, this means various objects can share the same method, but also have the ability to override shared methods with a more specific implementation

Inheritance

- The concept of inheritance: taking the features of one object and then adding some new features. In OOP, this means we can take an object that already exists and inherit all its properties and methods. We can then improve its functionality by adding new properties and methods.

Constructor functions

- An alternative way to create objects

```
const Dice = function(sides=6){    this.sides = sides;    this.roll =
function() {        return Math.floor(this.sides * Math.random() + 1)
}}

Then create an instance:

const redDice = new Dice();

<< Dice { sides: 6, roll: [Function] }
```

Static methods are not available to instances of the class. So, in our example, the instances of Dice such as redDice and blueDice cannot call the static description() method

Prototype

The prototype can be used to add any new properties and methods after the class has been declared. It should be used to define any properties that will remain the same for every instance of the class.

Public and Private Methods

By default, an object's methods are public in JavaScript. Methods and properties are said to be public because they can be queried directly and changed by assignment. The dynamic nature of the language means that an object's properties and methods can be changed after it has been created.

giving users or external services too much access to properties and methods could be a recipe for disaster!

Chapter Summary

- Object-oriented programming (OOP) is a way of programming that uses objects that encapsulate their properties and methods.

- The main concepts of OOP are encapsulation, polymorphism, and inheritance.
- Constructor functions can be used to create instances of objects.
- ES6 introduced class declarations that use the class keyword. These can be used in place of constructor functions.
- Inside a constructor function or class declaration, the keyword refers to the object returned by the function.
- All instances of a class or constructor function inherit all the properties and methods of its prototype.
- The prototype is live, so new properties and methods can be added to existing instances.
- The prototype chain is used to find an available method. If an object lacks a method, JavaScript will check whether its prototype has the method. If not, it will check that function's prototype until it finds the method or reaches the Object constructor function.
- Private properties and methods can be created by defining variables using const and defining a function inside a constructor function. These can be made public using getter and setter functions.
- Monkey-patching is the process of adding methods to built-in objects by augmenting their prototypes. This should be done with caution as it can cause unexpected behavior in the way built-in objects work.
- A mixin method can be used to add properties and methods from other objects without creating an inheritance chain.
- Methods can be chained together and called in sequence if they return a reference to this.
- Polymorphism allows objects to override shared methods with a more specific implementation.
- The value of this is not retained inside nested functions, which can cause errors. This can be worked around by using that = this, using the bind(this) method, and using arrow functions.

- Methods can be borrowed from other objects.

- Composition over inheritance is a design pattern where objects are composed of 'building-block' objects, rather than inheriting all their properties and methods from a parent class.

Chapter 15

Libraries

- A big advantage of utilizing a popular library is that it will be used by lots of people and thoroughly tested. It will most likely have been optimized and battle-tested for nearly every eventuality.

- There are some disadvantages to using libraries, however. You need to include the code for the library as well as your code. This increases the amount of code that needs to be downloaded by a website, which in some cases can cause performance issues.

- Be careful with using libraries they can be a big help or a hindrance.

- Don't rely on libraries.

Modular JavaScript

- A module is a self-contained piece of code that provides functions and methods that can then be used in other files and by other modules

MVC Framework

- Models are objects that implement the functionality for creating, reading, updating, and deleting (known as CRUD tasks) specific pieces of information about the application, as well as any other associated logic and behavior.
- Views provide a visual representation of the model showing all the relevant information.
- Controllers link models and views together by communicating between them
- JS is more considered an MV* framework, where they don't use controllers as much most of the time.

Chapter Summary

- JavaScript libraries provide methods to make common tasks easier to achieve.

- Libraries can make programming much easier, but you should think carefully about whether you require a library, and which one is best for your needs.

- jQuery and Lodash are two popular libraries that provide a large number of useful and well-tested functions.

- npm and Yarn are package managers that can be used to install JavaScript packages, as well as any dependencies that they require.

- A module is a self-contained piece of code that provides functions and methods that can then be used in other files and by other modules.

- ES6 added support for modules, allowing code to be abstracted into their self-contained files and imported into another file.

- The MVC pattern is used to organize code into distinct sections that are responsible for different elements of an application.

- Template files can be used to separate view code from JavaScript; they also enable dynamic code and programming logic to be used to generate markup.

- React and Vue.js are popular JavaScript view libraries that render components and keep track of their state.

- Minification is the process of removing any redundant characters from the code to reduce its file size.

- Files can be compressed on the server using the gzip compression tool.

- Webpack can be used to bundle multiple files into a single bundle, and automate common tasks such as transpiring, minifying code, and running tests.

- Before code is deployed, it should be concatenated into a single file, minified, and compressed. The script tag should be placed just before the closing </body> tag to ensure that all elements on the page have loaded before the script runs.