WEBDEVELOPERMENT

FRONT END DEVELOPER

JULY 25, 2021

What is HTML?

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

All HTML documents must start with a document type declaration: `<!DOCTYPE html>`.

The HTML document itself begins with `<html>` and ends with `</html>`.

The visible part of the HTML document is between `<body>` and `</body>`.

A Simple HTML Document

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

CSS

CSS (Cascading Style Sheets) is the code that styles web content. *CSS basics* walks through what you need to get started. We'll answer questions like: How do I make text red? How do I make content display at a certain location in the (webpage) layout? How do I decorate my webpage with background images and colors?

What is CSS?

Like HTML, CSS is not a programming language. It's not a markup language either. **CSS is a style sheet language.** CSS is what you use to selectively style HTML elements. For example, this CSS selects paragraph text, setting the color to red:

Why Use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

```
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
```

CSS preprocessor
A **CSS preprocessor** is a program that lets you generate CSS from the preprocessor's own unique syntax. There are many CSS preprocessors to choose from, however most CSS preprocessors will add some features that don't exist in pure CSS, such as mixin, nesting selector, inheritance selector, and so on. These features make the CSS structure more readable and easier to maintain.

To use a CSS preprocessor, you must install a CSS compiler on your web server; Or use the CSS preprocessor to compile on the development environment, and then upload compiled CSS file to the web server.

Why Learn Sass?

Sass is an easy-to-use styling language that helps reduce a lot of the repetition and maintainability challenges of traditional CSS. Learning Sass will not only let you scale styles when working on big web development projects, it will also make it much faster and more efficient to write reusable styles from scratch for smaller projects.

Take-Away Skills:

This course will teach you how to use nesting, variables, mixins, placeholders, and functions to write more expressive and reusable styles. At the end of the

course, you will be able to transition a CSS codebase to SCSS and style multiple websites.

Note on Prerequisites:

We recommend you complete Learn CSS, particularly the basics of working with CSS selectors and visual rules, before taking this course.

JAVASCRIPT

javaScript is a scripting or programming language that allows you to implement complex features on web pages — every time a web page does more than just sit there and display static information for you to look at — displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes, etc. — you can bet that JavaScript is probably involved. It is the third layer of the layer cake of standard web technologies, two of which (HTML and CSS) we have covered in much more detail in other parts of the Learning Area.

What Is React?

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called "components".
React has a few different kinds of components, but we'll start with React.Component subclasses:

```
class ShoppingList extends React.Component {
  render() {
    return (
      <div className="shopping-list">
        <h1>Shopping List for {this.props.name}</h1>
        <ul>
          <li>Instagram</li>
          <li>WhatsApp</li>
          <li>Oculus</li>
        </ul>
      </div>
    );
  }
}

// Example usage: <ShoppingList name="Mark" />
```

We'll get to the funny XML-like tags soon. We use components to tell React what we want to see on the screen. When our data changes, React will efficiently update and re-render our components.

Here, ShoppingList is a **React component class**, or **React component type**. A component takes in parameters, called props (short for "properties"), and returns a hierarchy of views to display via the render method.
The render method returns a *description* of what you want to see on the screen. React takes the description and displays the result. In particular, render returns a **React element**, which is a lightweight description of what to render. Most React developers use a special syntax called "JSX" which makes these structures easier to write. The syntax is transformed at build time to React.createElement('div'). The example above is equivalent to:

```
return React.createElement('div', {className: 'shopping-list'},
  React.createElement('h1', /* ... h1 children ... */),
  React.createElement('ul', /* ... ul children ... */)
);
```

See full expanded version.
If you're curious, createElement() is described in more detail in the API reference, but we won't be using it in this tutorial. Instead, we will keep using JSX.
JSX comes with the full power of JavaScript. You can put *any* JavaScript expressions within braces inside JSX. Each React element is a JavaScript object that you can store in a variable or pass around in your program.
The ShoppingList component above only renders built-in DOM components like and . But you can compose and render custom React components too. For example, we can now refer to the whole shopping list by writing <ShoppingList />. Each React component is encapsulated and can operate independently; this allows you to build complex UIs from simple components.

Vue.js Directives

Vue.js uses double braces **{{ }}** as place-holders for data.

Vue.js directives are HTML attributes with the prefix **v-**

Vue Example

In the example below, a new Vue object is created with **new Vue()**.

The property **el:** binds the new Vue object to the HTML element with **id="app"**.

Example

```
<div id="app">
<h1>{{ message }}</h1>
</div>

<script>
```

```
var myObject = new Vue({
   el: '#app',
   data: {message: 'Hello Vue!'}
})

</script>
```

Try it Yourself »

AngularJS Directives

AngularJS uses double braces {{ }} as place holders for data.

AngularJS directives are HTML attributes with the prefix ng-

The ng-app directive initializes an AngularJS application.

The ng-init directive initializes application data.

Example

```
<div ng-app="" ng-init="message='Hello AngularJS!'">
  <h1>{{ message }}</h1>
</div>
```

Try it Yourself »

Node.js is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside of a web browser. Node.js is a popular, lightweight web framework for beginners, and it is used by many big companies like Netflix and Uber.

When we typically think of JavaScript, our mind tends to go to web development. Until Node.js came along, there was really no way to run JavaScript outside of a browser. When we write a backend server and database, Node.js is a popular choice because we can run our code as a standalone application rather than something that can only be evaluated in a browser environment.

Node.js is an important tool for any JavaScript developer to understand. So, today, we'll introduce you to Node.js and show you how to get started with a project.

TYPESCRIPT

By definition, "TypeScript is JavaScript for application-scale development."

TypeScript is a strongly typed, object oriented, compiled language. It was designed by **Anders Hejlsberg** (designer of C#) at Microsoft. TypeScript is both a language and a set of tools. TypeScript is a typed superset of JavaScript compiled to JavaScript. In other words, TypeScript is JavaScript plus some additional features.

JQUERY

JQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

REDUX

Redux is simply a store to store the state of the variables in your app. Redux creates a process and procedures to interact with the store so that components will not just update or read the store randomly. Similar to the bank. It does not mean because you have money in the bank that you can go anytime, open the vault, and take money. You have to go through certain steps to withdrawal money.

TASK RUNNER

What are task runners?

Task runners are **the heroes** (or villains, depending on your point of view) that quietly toil behind most web and mobile applications. Task runners provide value through the automation of numerous development tasks such as concatenating files, spinning up development servers and compiling code.

GRANT

JavaScript developer Ben Alman launched the command task runner Grunt, which he has been managing and developing with a small team ever since. The program's code, which is available under the **MIT license**, can be downloaded from the official website and is also available to use for free at GitHub. As with other comparable tools, Grunt is based on the JavaScript runtime environment, Node.js, and, thanks to **grunt-cli**, offers its own command line interface, which can be installed via the **Node Package Manager** like the task runner itself

GULP

Gulp is a free task runner, initially launched in July 2013 by American software company Fractal Innovations in collaboration with the GitHub community. As with Grunt, the program is available under an open source **MIT license**. It is based on the JavaScript platform **Node.js** and, like its competitor, uses the **npm** package manager. Regarding the structure, Grunt and Gulp are relatively similar to one another; Gulp is also a command line tool, so it has a suitable user interface with **gulp-cli**. The *package.json* configuration file and the gulpfile (*gulpfile.js*), which lists possible tasks, are also usually used. If both are added to the web directory, the task runner can be used for **workflow optimization**.

Webpack is a **module bundler**. Webpack can take care of bundling alongside a separate task runner. However, the line between bundler and task runner has become blurred thanks to community-developed webpack plugins. Sometimes these plugins are used to perform tasks that are usually done outside of webpack, such as cleaning the build directory or deploying the build although you can defer these tasks outside of webpack.

HPM

What is HPM ?

There may be more than one meaning of **HPM** , so check it out all meanings of **HPM** one by one.

HPM definition / HPM means?

The Definition of HPM is given above so check it out related information.

What is the meaning of HPM ?

The meaning of the **HPM** is also explained earlier. Till now you might have got some idea about the acronym, abbreviation or meaning of **HPM** . What does **HPM** mean? is explained earlier. You might also like some similar terms related to **HPM** to know more about it. This site contains various terms

related to bank, Insurance companies, Automobiles, Finance, Mobile phones, software, computers,Travelling, School, Colleges, Studies, Health and other terms.

BOWER

Web sites are made of lots of things — frameworks, libraries, assets, and utilities. Bower manages all these things for you.

Keeping track of all these packages and making sure they are up to date (or set to the specific versions you need) is tricky. Bower to the rescue!

Bower can manage components that contain HTML, CSS, JavaScript, fonts or even image files. Bower doesn't concatenate or minify code or do anything else - it just installs the right versions of the packages you need and their dependencies.

To get started, Bower works by fetching and installing packages from all over, taking care of hunting, finding, downloading, and saving the stuff you're looking for. Bower keeps track of these packages in a manifest file, bower.json. How you use packages is up to you. Bower provides hooks to facilitate using packages in your tools and workflows.

Bower is optimized for the front-end. If multiple packages depend on a package - jQuery for example - Bower will download jQuery just once. This is known as a flat dependency graph and it helps reduce page load.

Install Bower

Bower is a command line utility. Install it with npm.

```
$ npm install -g bower
```

Bower requires node, npm and git.

GIT COMMAND LINE

The Command Line

There are a lot of different ways to use Git. There are the original command-line tools, and there are many graphical user interfaces of varying capabilities. For this book, we will be using Git on

the command line. For one, the command line is the only place you can run **all** Git commands — most of the GUIs implement only a partial subset of Git functionality for simplicity. If you know how to run the command-line version, you can probably also figure out how to run the GUI version, while the opposite is not necessarily true. Also, while your choice of graphical client is a matter of personal taste, **all** users will have the command-line tools installed and available.