Other Internet Protocols

Hypertext Transfer Protocols (HTTP) are used on top of Transmission Control Protocol (TCP) to transfer webpages and other content from websites.
This reading explores other protocols commonly used on the Internet.

Dynamic Host Configuration Protocol (DHCP)

You've learned that computers need IP addresses to communicate with each other. When your computer connects to a network, the Dynamic Host Configuration Protocol or DHCP as it is commonly known, is used to assign your computer an IP address.
Your computer communicates over User Datagram Protocol (UDP) using the protocol with a type of server called a DHCP server. The server keeps track of computers on the network and their IP addresses. It will assign your computer an IP address and respond over the protocol to let it know which IP address to use. Once your computer has an IP address, it can communicate with other computers on the network.

Domain Name System Protocol (DNS)

Your computer needs a way to know with which IP address to communicate when you visit a website in your web browser, for example, **`meta.com`**. The Domain Name System Protocol, commonly known as DNS, provides this function. Your computer then checks with the DNS server associated with the domain name and then returns the correct IP address.

Internet Message Access Protocol (IMAP)

Do you check your emails on your mobile or tablet device? Or maybe you use an email application on your computer?
Your device needs a way to download emails and manage your mailbox on the server storing your emails. This is the purpose of the Internet Message Access Protocol or IMAP.

Simple Mail Transfer Protocol (SMTP)

Now that your emails are on your device, you need a way to send emails. The Simple Mail Transfer Protocol, or SMTP, is used. It allows email clients to submit emails for sending via an SMTP server. You can also use it to receive emails from an email client, but IMAP is more commonly used.

Post Office Protocol (POP)

The Post Office Protocol (POP) is an older protocol used to download emails to an email client. The main difference in using POP instead of IMAP is that POP will delete the emails on the server once they have been downloaded to your local device. Although it is no longer commonly used in email clients, developers often use it to implement email automation as it is a more straightforward protocol than IMAP.

File Transfer Protocol (FTP)

When running your websites and web applications on the Internet, you'll need a way to transfer the files from your local computer to the server they'll run on. The standard protocol used for this is the File Transfer Protocol or FTP. FTP allows you to list, send, receive and delete files on a server. Your server must run an FTP Server and you will need an FTP Client on your local machine. You'll learn more about these in a later course.

Secure Shell Protocol (SSH)
When you start working with servers, you'll also need a way to log in and interact with the computer remotely. The most common method of doing this is using the Secure Shell Protocol, commonly referred to as SSH. Using an SSH client allows you to connect to an SSH server running on a server to perform commands on the remote computer.
All data sent over SSH is encrypted. This means that third parties cannot understand the data transmitted. Only the sending and receiving computers can understand the data.

SSH File Transfer Protocol (SFTP)
The data is transmitted insecurely when using the File Transfer Protocol. This means that third parties may understand the data that you are sending. This is not right if you transmit company files such as software and databases. To solve this, the SSH File Transfer Protocol, alternatively called the Secure File Transfer Protocol, can be used to transfer files over the SSH protocol. This ensures that the data is transmitted securely. Most FTP clients also support the SFTP protocol.

----

https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
https://www.amazon.com/Introduction-Networking-How-Internet-Works/dp/1511654945/
https://developer.chrome.com/docs/devtools/overview/
https://firefox-source-docs.mozilla.org/devtools-user/index.html
https://code.visualstudio.com/docs

HTML

https://developer.mozilla.org/en-US/docs/Web/HTML/Element
https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form
https://www.w3.org/TR/WD-DOM/introduction.html
https://w3c.github.io/html-aria/
https://www.w3.org/WAI/ARIA/apg/

CSS

Different types of selectors
When styling a web page, there are many types of selectors available that allow developers to be as broad or as specific as they need to be when selecting HTML elements to apply CSS rules to.

Here you will learn about some of the common CSS selectors that you will use as a developer.

Element Selectors

The element selector allows developers to select HTML elements based on their element type. For example, if you use **`p`** as the selector, the rule will apply to all **`p`** elements on the webpage.

HTML

```html
<p>In a hidden land...</p>
```

CSS

```css
p {
  color: blue;
}
```

ID Selectors

The ID selector uses the id attribute of an HTML element. Since the id is unique within a webpage, it allows the developer to select a specific element for styling. ID selectors are prefixed with a **`#`** character.

HTML

```html
<span id="latest">New!</span>
```

CSS

```css
#latest {
  background-color: purple;
}
```

Class Selectors

Elements can also be selected based on the class attribute applied to them. The CSS rule has been applied to all elements with the specified class name. The class selector is prefixed with a **`.`** character.

In the following example, the CSS rule applies to both elements as they have the **`navigation`** CSS class applied to them.

HTML

```
<a class="navigation">Go Back</a>
<p class="navigation">Go Forward</p>
```

CSS

```
.navigation {
  margin: 2px;
}
```

Element with Class Selector

A more specific method for selecting HTML elements is by first selecting the HTML element, then selecting the CSS class or ID.

The example below selects all `p` elements that have the CSS class `introduction` applied to them.

HTML

```
<p class="introduction"></p>
```

CSS

```
p.introduction {
  margin: 2px;
}
```

Descendant Selectors

Descendant selectors are useful if you need to select HTML elements that are contained within another selector.

Let's explore an example.

You have the following HTML structure and CSS rule.

HTML

```html
<div id="blog">
  <h1>Latest News</h1>
  <div>
    <h1>Today's Weather</h1>
    <p>The weather will be sunny</p>
  </div>
  <p>Subscribe for more news</p>
</div>
<div>
  <h1>Archives</h1>
</div>
```

CSS

```css
#blog h1 {
  color: blue;
}
```

The CSS rule will select all **`h1`** elements that are contained within the element with the ID **`blog`**. The CSS rule will not apply to the **`h1`** element containing the text **`Archives`**.
The structure of a descendant selector is a CSS selector, followed by a single space character, followed by another CSS selector.
Multiple descendants can also be selected. For example, to select all **`h1`** elements that are descendants of **`div`** elements which are descendants of the **`blog`** element, the selector is specified as follows.
CSS

```
#blog div h1 {
  color: blue;
}
```

Child Selectors

Child selectors are more specific than descendant selectors. They only select elements that are immediate descendants (children) of a selector (the parent).

For example, you have the following HTML structure:

HTML

```
<div id="blog">
  <h1>Latest News</h1>
  <div>
    <h1>Today's Weather</h1>
    <p>The weather will be sunny</p>
  </div>
  <p>Subscribe for more news</p>
</div>
```

If you wanted to style the **`h1`** element containing the text **`Latest News`**, you can use the following child selector:

CSS

```
#blog > h1 {
  color: blue;
}
```

This will select the element with the ID **`blog`** (the parent), then it will select all **`h1`** elements that are contained directly in that element (the children). The structure of the child selector is a CSS selector

followed by the child combinator symbol **`>`** followed by another CSS selector.
Note that this will not go beyond a single depth level. Therefore, the CSS rule will **not** be applied to the **`h1`** element containing the text **`Today's Weather`**.
:hover Pseudo-Class
A special keyword called a pseudo-class allows developers to select elements based on their state. Don't worry too much about what that means right now. For now, let's look at how the hover pseudo-class allows you to style an element when the mouse cursor hovers over the element.
The simplest example of this is changing the color of a hyperlink when it is hovered over. To do this, you add the **`:hover`** pseudo-class to the end of the selector. In the following example, adding **`:hover`** to the **`a`** element will change the color of the hyperlink to orange when it is hovered over.
CSS

```
a:hover {
  color: orange;
}
```

This pseudo-class is very useful for creating visual effects based on user interaction.
Other Selectors
There are many other CSS selectors available to style your webpage.

Text and color in CSS
As you design websites, you'll be working a lot with colors and text. There are many different ways to display text and equally as many ways to define colors.
This reading covers how text and color work in CSS.
Color
Colors are used in many CSS properties, for example:

```
}
p {
  color: blue;
```

From CSS Version 3, there are five main ways to reference a color.

- By RGB value,
- By RGBA value,
- By HSL value,
- By hex value and
- By predefined color names.

RGB value

RGB is a color model that adds the colors red (R), green (G) and blue (B) together to create colors. This is based on how the human eye sees colors.
Each value is defined as a number between **`0`** and **`255`**, representing the intensity of that color.
For example, the color red would have the RGB value of **`255,0,0`** since the intensity of the red color would be 100% while blue and green would be 0%.
The color black then would be **`0,0,0`** and the color white **`255,255,255`**.
When using RGB values in CSS, they can be defined using the **`rgb`** keyword:

```
p {
  color: rgb(255, 0, 0);
}
```

RGBA value

RGBA is an extension of RGB that add an alpha (A) channel. The alpha channel represents the opacity, or transparency, of the color.
Similar to RGB, this is specified in CSS using the **`rgba`** keyword:

```
p {
  color: rgba(255, 0, 0, 0.8);
}
```

HSL value

HSL is a newer color model defined as Hue (H), Saturation (S) and Lightness (L). The aim of the model is to simplify mental visualization of the color that the value represents.
Think of a rainbow that has been turned into a full circle. This represents the Hue. The Hue value is the degree value on this circle, from 0 degrees to 360 degrees. 0 is red, 120 is green and 240 is blue.

Saturation is the distance from the center of the circle to its edge. The saturation value is represented by a percentage from 0% to 100% where 0% is the center of the circle and 100% is its edge. For example, 0% will mean that the color is more grey and 100% represents the full color.
Lightness is the third element of this color model. Think of it as turning the circle into a 3D cylinder where the bottom of the cylinder is more black and toward the top is more white. Therefore, lightness is the distance from the bottom of the cylinder to the top. Again, lightness is represented by a percentage from 0% to 100% where 0% is the bottom of the cylinder and 100% is its top. In other words, 0% will mean that the color is more black and 100% is white.

In CSS, you use the **`hsl`** keyword to define a color with HSL.

```
}
p {
  color: hsl(0, 100%, 50%);
```

Hex value

Colors can be specified using a hexadecimal value. If you're unfamiliar with hexadecimal, think of it as a different number set.

Decimal is what you use every day. Digits range from `0` to `9` before tens and hundreds are used.

Hexadecimal is similar, except it has 16 digits. This is counted as **`0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F`**.

In fact, you can convert between decimal and hexadecimal. Decimal **`10`** is equal to hexadecimal **`A`**. Hexadecimal **`F`** is equal to decimal **`15`**.

Hexadecimal can also go to tens and hundreds. For example, decimal **`16`** is equal to hexadecimal **`10`**, with **`10`** being the next number after **`F`**.

It can be a little confusing at first but don't worry, there are plenty of converters available if you get stuck.

Colors specified using hexadecimal are prefixed with a **`#`** symbol followed by the RGB value in hexadecimal format.

For example, the color red which is RGB **`255,0,0`** would be written as hexadecimal **`#FF0000`**.

Again don't worry if you get stuck, there are plenty of converters available for this too!

Predefined color names

Modern web browsers support 140 predefined color names. These color names are for convenience purposes and can be mapped to equivalent hex/RGB/HSL values.

Some common color names available are listed below.

```
black
silver
gray
white
maroon
red
purple
fuchsia
green
lime
olive
yellow
navy
blue
teal
aqua
```

Text

With CSS there are many ways to change how text is displayed. In this section, you'll learn the most common text manipulation CSS properties.

Text Color

The **`color`** property sets the color of text. The following CSS sets the text color for all paragraph elements to red.

```
p {
  color: red;
}
```

Text Font and Size

There are many different fonts to display text on your computer. In simple terms, a font is a collection of text characters written in a specific style and size.

If you've used a word processor before, you're probably familiar with the fonts Times New Roman and Calibri.

To set the font used by text in CSS you use the **`font-family`** property.

```
}
p {
  font-family: "Courier New", monospace;
```

Since computers vary in what fonts they have installed, it is recommended to include several fonts when using the **`font-family`** property. These are specified in a fallback order, meaning that if the first font is not available, it will check for the second font. If the second font is not available, then it will check for the third font and so on. If none of the fonts are available, it will use the browser's default font.

To set the size of the font, the **`font-size`** property is used.

```
p {
  font-family: "Courier New", monospace;
  font-size: 12px;
}
```

Text Transformation

Text transformation is useful if you want to ensure the correct capitalization of the text content. In the example below, the CSS rule will change all text in paragraph elements to uppercase using the **`text-transform`** property:

```
p {
  text-transform: uppercase;
}
```

The most commonly used values for the **`text-transform`** property are: **`uppercase`**, **`lowercase`**, **`capitalize`** and **`none`**. The default value used is **`none,`** which means the text displays as it was written in the HTML document.

Text Decoration

The **`text-decoration`** property is useful to apply additional decoration to text such as underlining and line-through (strikethrough).

```
p {
  text-decoration: underline;
}
```

It is possible to set the color, thickness and styling of the decoration too. In the example below, the underline will be a solid red line that is 5 pixels thick.

```
}
p {
  text-decoration: underline red solid 5px;
```

If this is confusing, don't worry. These properties can be individually set using the **`text-decoration-line`**, **`text-decoration-color`**, **`text-decoration-style`** and **`text-decoration-thickness`** properties. Let's use the same example again and define it using the individual properties:

```
}
  text-decoration-color: red;
  text-decoration-style: solid;
  text-decoration-thickness: 5px;
p {
  text-decoration-line: underline;
```

The most common **`text-decoration-line`** values used are: **`underline`**, **`overline`**, **`line-through`** and **`none`**. None is the default value to use no text decoration.

There are many styles available for the **`text-decoration-style`** property; **`solid`**, **`double`**, **`dotted`**, **`dashed`** and **`wavy`**. The **`text-decoration-style`** property requires the decoration line to be defined. If the decoration style is not specified, **`solid`** will be used.

Alignment basics

Let's explore how to align text and HTML elements using CSS.

Let's first focus on horizontal alignment. Vertical alignment is more difficult so you'll explore that later on.

Text Alignment

Aligning text within an HTML element is very simple. To do this, you use the **`text-align`** CSS property. In the following example, the CSS rule is setting the text of all paragraph elements to be center aligned.

```
}
p {
    text-align: center;
```

Text alignment can be set to **`left`**, **`right`**, **`center`** and **`justify`**.

The **`justify`** alignment spreads the text out so that every line of the text has the same width.

The default alignment is **`left`** for languages that are left-to-right such as English. For right-to-left languages such as Arabic, the default alignment is **`right`**.

HTML Element Alignment

HTML element alignment is more complicated than text alignment. To align HTML elements, you must consider the box model and document flow from previous lessons. Aligning an HTML element is done by changing the properties of its box model and how it impacts the document flow.

HTML Element Center Alignment

To center align an element, you set a width on the element and push its margins out to fill the remaining available space of the parent element as in the following HTML structure:

```
</div>
<div class="parent">
  <div class="child">
  </div>
```

In your CSS, you'll set the **`parent`** element to have a red border to visualize the space it occupies:

```
}
```

```
.parent {
  border: 4px solid red;
```

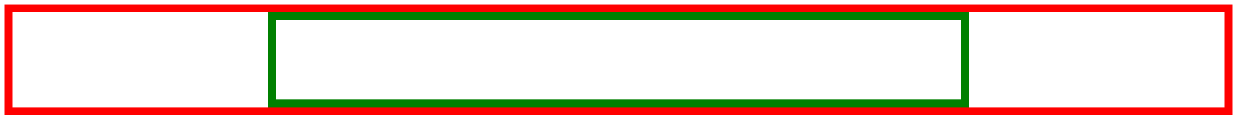
The **`child`** element will have a width equal to 50% of the **`parent`** element with a padding of 20 pixels. Note that **`padding: 20px`** is shorthand for setting the padding top, bottom, left and right to **`20px`**. To visualize the space it occupies, set the border to green:

```
.child {
  width: 50%;
  padding: 20px;
  border: 4px solid green;
}
```

To align the element to the center, set its **`margin`** property to **`auto`**. The **`auto`** will tell the browser to calculate the margin automatically based on the space available.

```
.child {
  width: 50%;
  padding: 20px;
  border: 4px solid green;
  margin: auto;
}
```

The result is the **`child`** element is centered within the **`parent`** element:



It is important to note that this works because the **`div`** element is a block-level element. If you want to align an inline element like **`img`**, you will need to change it to a block-level element. Similar to the **`div`** example, you add the **`img`** to a parent element:

```
<div class="parent">
  <img src="photo.png" class="child">
</div>
```

The CSS rule then changes the **`img`** element to a block-level element and sets its margin to **`auto`**:

```
.child {
  display: block;
  width: 50%;
  margin: auto;
}
```

To be more precise, in CSS you can set only the left and right margins to auto. This allows you to set the top and bottom margins to specific values if needed.

```
.child {
  display: block;
  width: 50%;
  margin-left: auto;
  margin-right: auto;
}
```

HTML Element Left / Right Alignment

The two most common ways to left and right align elements are to use the **`float`** property and the **`position`** property.

The **`position`** property has several value options that impact how the element displays in the document flow. You'll explore how to use the **`position`** property later on. For now, let's focus on the **`float`** property.

The **`float`** property sets an element's position relative to the text content within a parent element. Text will wrap around the element.

In the following example, the image will be aligned to the right of the **`div`** element. The text content will wrap around the image:

HTML

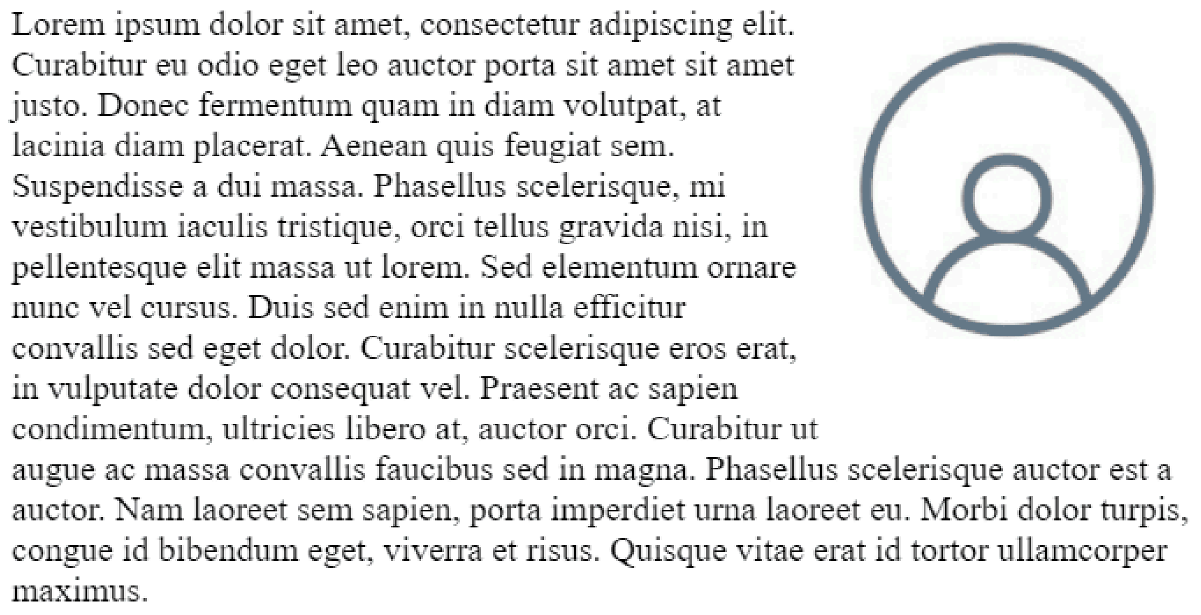
```
<div class="parent">
  <img src="photo.png" class="child"> Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Curabitur eu odio eget leo auctor porta sit
amet sit amet justo. Donec fermentum quam in diam volutpat, at lacinia
diam placerat. Aenean quis feugiat sem. Suspendisse a dui massa. Phasellus
scelerisque, mi vestibulum iaculis tristique, orci tellus gravida nisi, in
pellentesque elit massa ut lorem. Sed elementum ornare nunc vel cursus.
```

```
Duis sed enim in nulla efficitur convallis sed eget dolor. Curabitur
scelerisque eros erat, in vulputate dolor consequat vel. Praesent ac
sapien condimentum, ultricies libero at, auctor orci. Curabitur ut augue
ac massa convallis faucibus sed in magna. Phasellus scelerisque auctor est
a auctor. Nam laoreet sem sapien, porta imperdiet urna laoreet eu. Morbi
dolor turpis, congue id bibendum eget, viverra et risus. Quisque vitae
erat id tortor ullamcorper maximus.
</div>
```

CSS

```
.child {
  float: right;
}
```

The following displays in the web browser:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur eu odio eget leo auctor porta sit amet sit amet justo. Donec fermentum quam in diam volutpat, at lacinia diam placerat. Aenean quis feugiat sem. Suspendisse a dui massa. Phasellus scelerisque, mi vestibulum iaculis tristique, orci tellus gravida nisi, in pellentesque elit massa ut lorem. Sed elementum ornare nunc vel cursus. Duis sed enim in nulla efficitur convallis sed eget dolor. Curabitur scelerisque eros erat, in vulputate dolor consequat vel. Praesent ac sapien condimentum, ultricies libero at, auctor orci. Curabitur ut augue ac massa convallis faucibus sed in magna. Phasellus scelerisque auctor est a auctor. Nam laoreet sem sapien, porta imperdiet urna laoreet eu. Morbi dolor turpis, congue id bibendum eget, viverra et risus. Quisque vitae erat id tortor ullamcorper maximus.

https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
https://www.amazon.com/HTML-CSS-Design-Build-Websites/dp/1118008189/
https://www.amazon.com/CSS-Definitive-Guide-Visual-Presentation/dp/1449393195/

BOOTSTRAP

https://getbootstrap.com/docs/5.3/getting-started/introduction/

Other CSS frameworks and libraries
As a developer, you'll use many CSS libraries and frameworks throughout your career. As you move on to different projects and as technologies advance, knowing what solutions are available is critical. While Bootstrap is one of the most popular CSS libraries, many others are available, each with different purposes, designs and technical approaches. This reading will introduce you to other popular CSS libraries and frameworks.

Foundation
https://get.foundation/
Foundation is a framework for building user interfaces similar to Bootstrap. It is used by many large companies such as Pixar, Polar and Sonos. One prominent feature of Foundation is that it can be used to style content for sending via email.

Pure.css
https://pure-css.github.io/
Pure.css is another library for building user interfaces. While it doesn't have as many features as Bootstrap, it is designed to be minimal in file size. Smaller file sizes improve loading times for web pages as there is less data to transfer from the web server. If your next project is focused on minimal loading time, this library is worth considering.

Tailwind CSS
https://tailwindcss.com/
Tailwind CSS is a CSS framework that uses a utility-based approach for its CSS rules. This means that the framework provides many CSS classes with a single purpose. For example, the CSS class pt-6 sets the padding-top CSS property to 6 pixels. This means that you can be precise in applying styling to your HTML without writing CSS. The advantage to this is that it is more flexible for customizing your webpage's design using the framework. However, the disadvantage is that if multiple developers are working on a project, it could lead to inconsistent design if the team is not strict on its design rules.

UIKit
https://getuikit.com/
UIKit is a lightweight CSS framework featuring a small set of responsive components. Its simple design allows developers to easily customize the style rules and visuals.

MVP.css
https://andybrewer.github.io/mvp/
MVP.css is a small CSS library that automatically styles HTML elements without needing to apply CSS classes to them. The library aims to allow a developer to quickly prototype a user interface without worrying about the final design, while still being visually appealing. MVP comes from the technical term Minimal Viable Product, a product with sufficient features to demo to customers or

other business stakeholders.

Conclusion
If you're curious to learn more about these frameworks, their websites feature set up guides, tutorials and documentation to get started. It is a good exercise to compare and contrast different libraries and frameworks to understand different workflows available to you as a developer.

https://getbootstrap.com/
https://www.amazon.com/Bootstrap-Foundations-Mr-Daniel-Foreman/dp/B0948GRS8W/
https://www.amazon.com/Responsive-Web-Design-HTML5-CSS/dp/1839211563/
https://themes.getbootstrap.com/

React
https://react.dev/
https://code.angularjs.org/1.0.8/docs/guide/directive#reasonsbehindthecompilelinkseparation
https://jsfiddle.net/uf3sr8L7/

The Virtual DOM

React builds a representation of the browser Document Object Model or DOM in memory called the virtual DOM. As components are updated, React checks to see if the component's HTML code in the virtual DOM matches the browser DOM. If a change is required, the browser DOM is updated. If nothing has changed, then no update is performed.
As you know, this is called the **reconciliation** process and can be broken down into the following steps:
Step 1: The virtual DOM is updated.
Step 2: The virtual DOM is compared to the previous version of the virtual DOM and checks which elements have changed.
Step 3: The changed elements are updated in the browser DOM.
Step 4: The displayed webpage updates to match the browser DOM.
As updating the browser DOM can be a slow operation, this process helps to reduce the number of updates to the browser DOM by only updating when it is necessary.
But even with this process, if a lot of elements are updated by an event, pushing the update to the browser DOM can still be expensive and cause slow performance in the web application.
The React team invested many years of research into solving this problem. The outcome of that research is what's known as the React Fiber Architecture.
The Fiber Architecture allows React to incrementally render the web page. What this means is that instead of immediately updating the browser DOM with all virtual DOM changes, React can spread the update over time. But what does "over time" mean?
Imagine a really long web page in the web browser. If the user scrolls to the bottom, the top of the

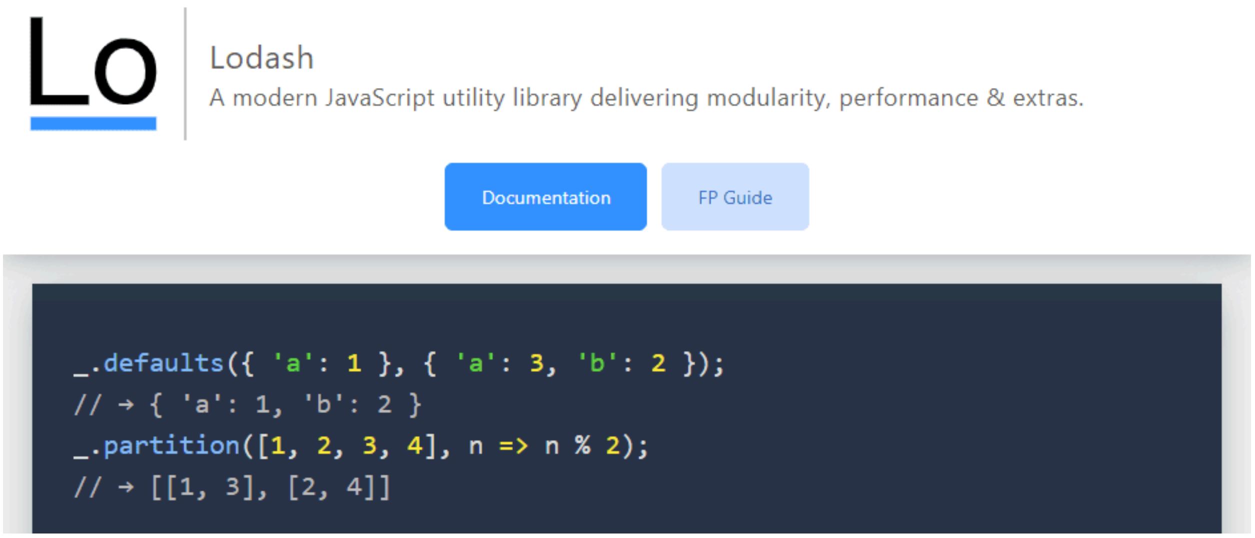
web page is no longer visible. The user then clicks a button on the bottom of the web page that updates some text on the top of the web page.
But the top of the page isn’t visible. Therefore, why update it immediately?
Perhaps there is text currently displayed on the bottom of the page that also updates when the button is clicked. Wouldn’t that be a higher priority to update than the non-visible text?
This is the principle of the React Fiber Architecture. React can optimize when and where updates occur to the browser DOM to significantly improve application performance and responsiveness to user input. Think of it as a priority system. The highest priority changes, the elements visible to the user, are updated first. While lower priority changes, the elements not currently displayed, are updated later.
While you’re unlikely to interact with the virtual DOM and Fiber Architecture yourself, it’s good to know what’s going on if issues occur during the development of your web application.
There are many tools available to help you investigate how React is processing your webpage. The official React Developer Tools web browser plugin developed by Meta will be one of the key tools in your developer toolbox. So, if you do have to look deeper into the code, you’ll have the right toolbox available to help you. These tools will be explored later on.

Alternatives to React

React is a library and not a framework. This means you'll often use other JavaScript libraries with it to build your application. In this reading, you will be briefly introduced to some JavaScript libraries commonly used with React.

Lodash

Official Website
As a developer, there's a lot of logic you'll commonly write across applications. For example, you might need to sort a list of items or round a number such as `3.14` to `3`. Lodash provides common logic such as these as a utility library to save you time as a developer.


Lo
Lodash
A modern JavaScript utility library delivering modularity, performance & extras.
Documentation
FP Guide

```
_.defaults({ 'a': 1 }, { 'a': 3, 'b': 2 });
// → { 'a': 1, 'b': 2 }
_.partition([1, 2, 3, 4], n => n % 2);
// → [[1, 3], [2, 4]]
```

Luxon

Official Website
You'll be working with dates and times often as a developer. Think of viewing a list of orders and when they were placed, or displaying a calendar schedule for an event. Dates and times are everywhere.
Luxon helps you work with dates and times by providing functions to manipulate and display them. For example, think of how dates are formatted in different countries. In the United States the format is **`Month Day Year`** but in Europe it is **`Day Month Year`**. This is one area where Luxon can help you display the date in the user's local format.

A powerful, modern, and friendly wrapper for JavaScript dates and times.

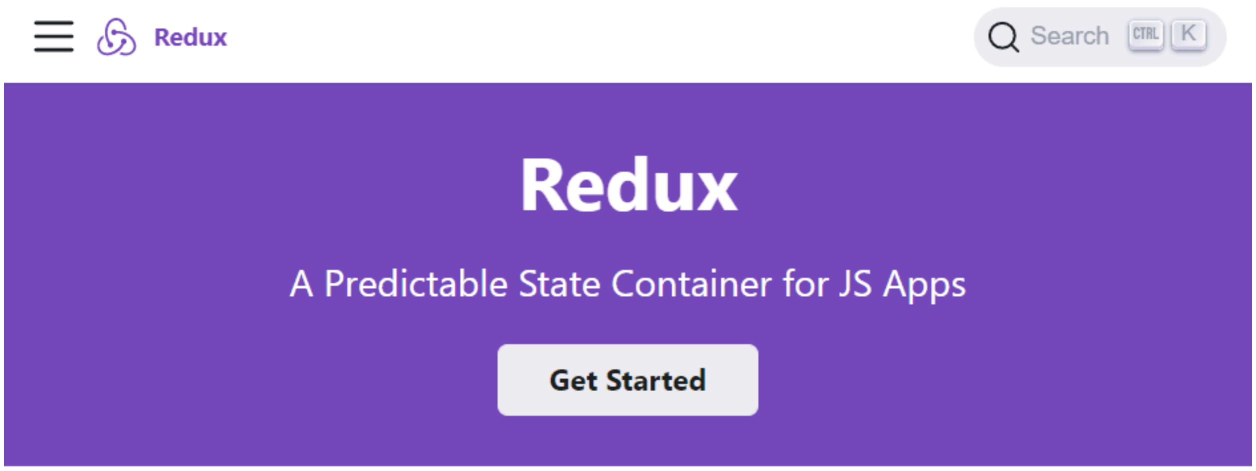
DateTimes, Durations, and Intervals

Immutable, chainable, unambiguous API.

Native time zone and Intl support (no locale or tz files)

Redux

Official Website
When building a web application, you'll need to keep track of its state. Think of when you shop online. The web application tracks items currently in your shopping cart. When you remove an item from the cart, the application needs to update what displays on the screen. This is where Redux comes in. It helps you manage your application state and even has advanced features such as undo and redo.

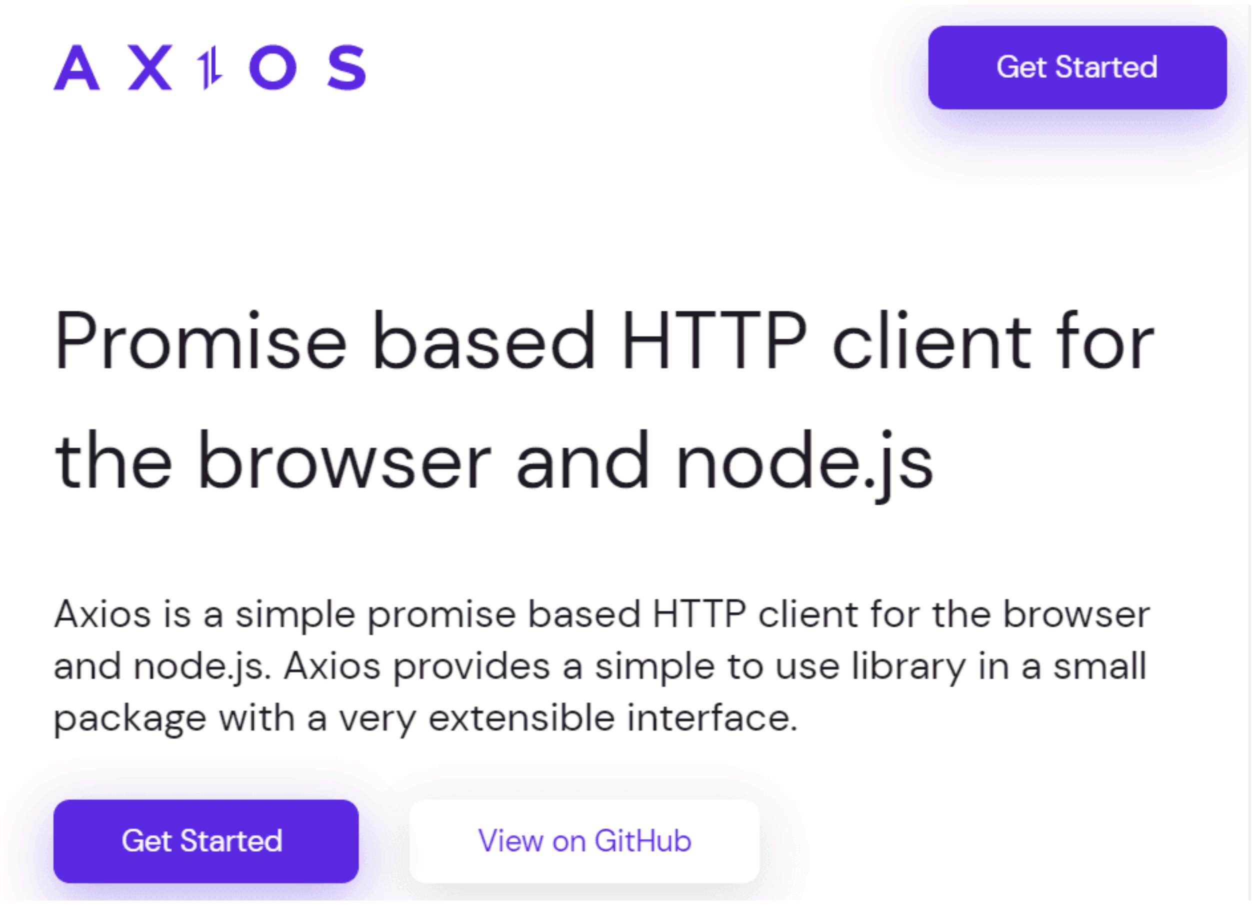


A Predictable State Container for JS Apps

Axios

Official Website

As a developer you'll be communicating with APIs over HTTP frequently. The Axios library helps to simplify sending HTTP requests and processing the response. It also provides advanced features allowing you to cancel requests and to change data received from the web server before your application uses the data.



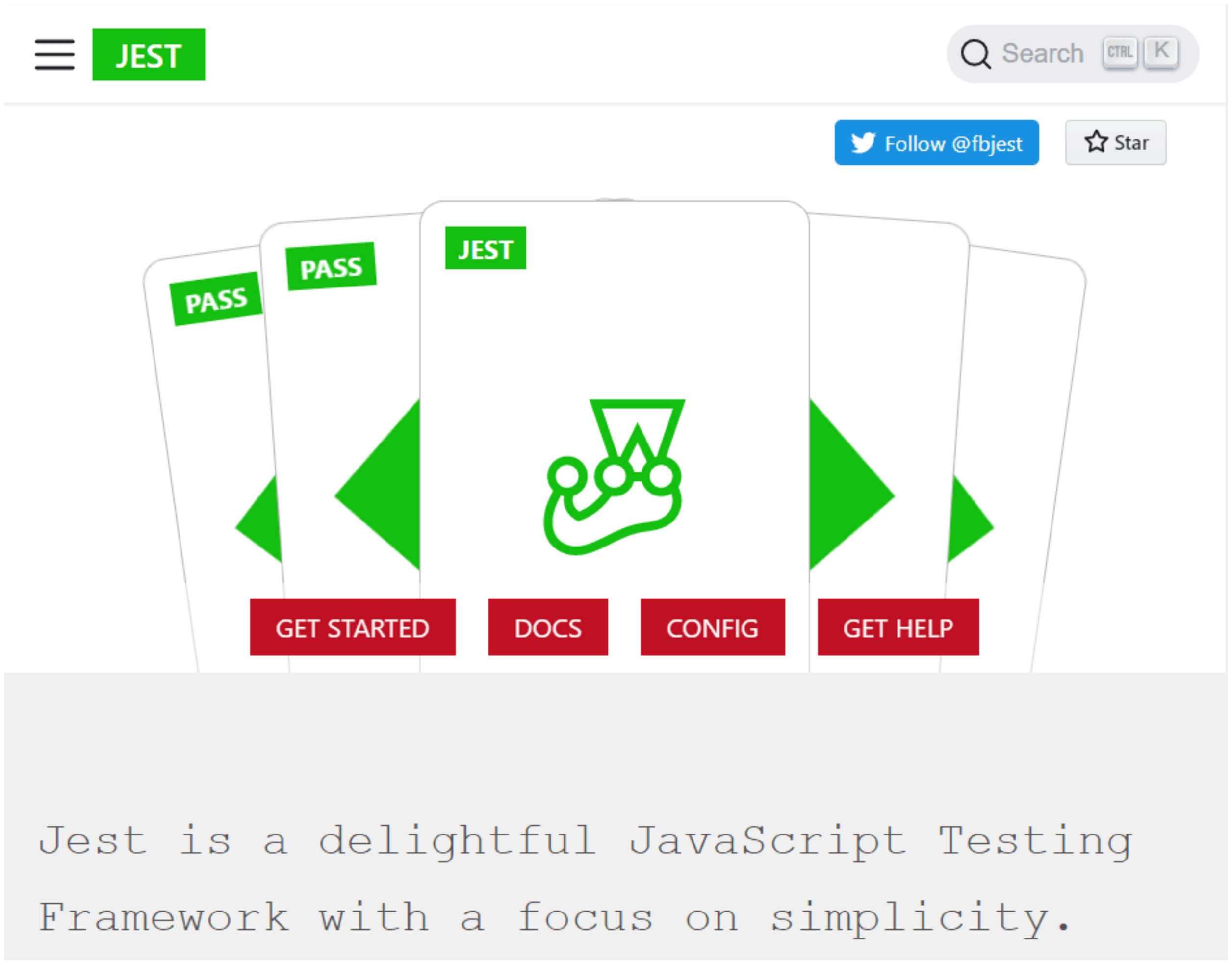
Promise based HTTP client for the browser and node.js

Axios is a simple promise based HTTP client for the browser and node.js. Axios provides a simple to use library in a small package with a very extensible interface.

Jest

Official Website

It is good practice to write automated tests for your code as a professional developer. The jest library helps you to do this and works with many libraries and frameworks. It also provides reporting utilities such as providing information on how much of your code is tested by your automated tests.



Jest is a delightful JavaScript Testing Framework with a focus on simplicity.

Conclusion

If you're curious to learn more about these libraries, their websites feature setup guides, tutorials and documentation to get started. These libraries will be covered later on.

https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-apps
https://github.com/facebook/react

thank you.

/tomo
/jcw