***Checkpoint 1***

* ***How Does The Web Work?***

The Web Is everywhere

We Use it more than we ever did before, also in many places where you might not see it.

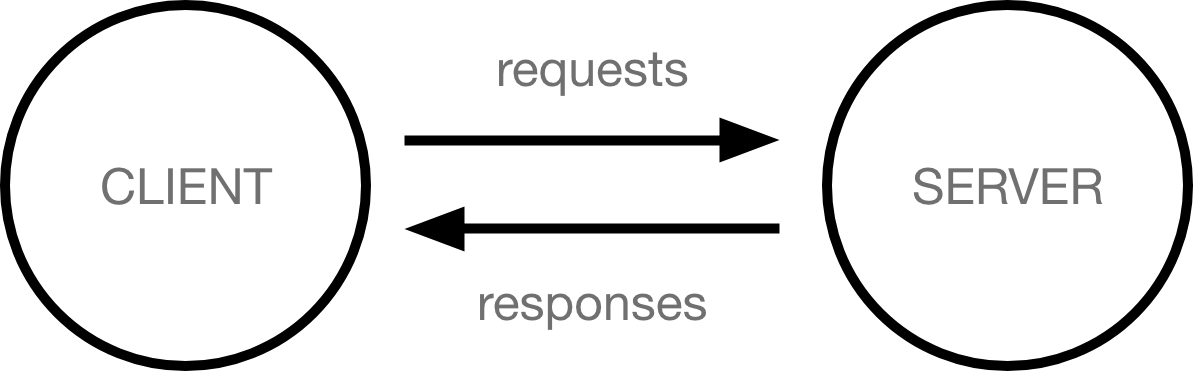
Because The Web is more than just websites you visit by entering a URL in your browser.

How the web works provides a simplified view of what happens when you view a webpage in a web browser on your computer or phone.

This theory is not essential to writing web code in the short term, but before long you'll really start to benefit from understanding what's happening in the background.

**+ Clients and servers:**

Computers connected to the web are called **clients** and **servers**. A simplified diagram of how they interact might look like this:



* Clients are the typical web user's internet-connected devices (for example, your computer connected to your Wi-Fi, or your phone connected to your mobile network) and web-accessing software available on those devices (usually a web browser like Firefox or Chrome).
* Servers are computers that store webpages, sites, or apps. When a client device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed in the user's web browser.
* [**The other parts of the toolbox**](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works#the_other_parts_of_the_toolbox)

In addition to the client and the server, we also need to say hello to:

* **Your internet connection**: Allows you to send and receive data on the web. It's basically like the street between your house and the shop.
* **TCP/IP**: Transmission Control Protocol and Internet Protocol are communication protocols that define how data should travel across the internet. This is like the transport mechanisms that let you place an order, go to the shop, and buy your goods. In our example, this is like a car or a bike (or however else you might get around).
* **DNS**: Domain Name Servers are like an address book for websites. When you type a web address in your browser, the browser looks at the DNS to find the website's real address before it can retrieve the website. The browser needs to find out which server the website lives on, so it can send HTTP messages to the right place (see below). This is like looking up the address of the shop so you can access it.
* **HTTP**: Hypertext Transfer Protocol is an application [protocol](https://developer.mozilla.org/en-US/docs/Glossary/Protocol) that defines a language for clients and servers to speak to each other. This is like the language you use to order your goods.
* **Component files**: A website is made up of many different files, which are like the different parts of the goods you buy from the shop. These files come in two main types:
  + **Code files**: Websites are built primarily from HTML, CSS, and JavaScript, though you'll meet other technologies a bit later.
  + **Assets**: This is a collective name for all the other stuff that makes up a website, such as images, music, video, Word documents, and PDFs.

[**So what happens, exactly?**](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works#so_what_happens_exactly)

When you type a web address into your browser (for our analogy that's like walking to the shop):

1. The browser goes to the DNS server, and finds the real address of the server that the website lives on (you find the address of the shop).
2. The browser sends an HTTP request message to the server, asking it to send a copy of the website to the client (you go to the shop and order your goods). This message, and all other data sent between the client and the server, is sent across your internet connection using TCP/IP.
3. If the server approves the client's request, the server sends the client a "200 OK" message, which means "Of course you can look at that website! Here it is", and then starts sending the website's files to the browser as a series of small chunks called data packets (the shop gives you your goods, and you bring them back to your house).
4. The browser assembles the small chunks into a complete web page and displays it to you (the goods arrive at your door — new shiny stuff, awesome!).

[**Order in which component files are parsed**](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works#order_in_which_component_files_are_parsed)

Once the client’s request is approved, the server first sends back the HTML (index) file — index.html is commonly named as such, as it is the first file of a website to be parsed by the server.

The HTML file can reference [CSS](https://developer.mozilla.org/en-US/docs/Learn/CSS) and [JavaScript](https://developer.mozilla.org/en-US/docs/Learn/JavaScript), either in external files via <link> and <script> elements respectively, or embedded in the HTML page via <style> and <script> elements.

From a server standpoint it is important to know the order in which these files are parsed when the response is sent back:

* The HTML file is parsed first and, by looking inside that file, the server recognises which CSS and JavaScript files are referenced.
* After the HTML has been parsed and a DOM tree structure has been generated from it, the linked CSS is then parsed, and styles are applied to the appropriate parts of the DOM tree. At this point, the visual representation of the page is painted to the screen, and the user sees the page.
* The JavaScript usually gets parsed and applied to the page after the HTML and CSS.

[**DNS explained**](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works#dns_explained)

Real web addresses aren't the nice, memorable strings you type into your address bar to find your favorite websites. They are special numbers that look like this: 63.245.215.20.

This is called an [IP address](https://developer.mozilla.org/en-US/docs/Glossary/IP_Address), and it represents a unique location on the web. However, it's not very easy to remember, is it? That's why Domain Name Servers were invented. These are special servers that match up a web address you type into your browser (like "mozilla.org") to the website's real (IP) address.

Websites can be reached directly via their IP addresses. You can find the IP address of a website by typing its domain into a tool like [IP Checker](https://ipinfo.info/html/ip_checker.php)

* ***How to become a web developer***

## What does a web developer do?

Also known as web programmers or web coders, web developers essentially make a website work by building the functionality, interactivity and visible structure of the site, normally based on the vision of designers and other key roles.

Web developers are also responsible for ensuring a site functions correctly on all browsers - both desktop and mobile - through testing. Once a site is live, a developer carries out  updates and other maintenance tasks as necessary.

## What qualifications do you need to become a web developer?

While there are no formal or specific qualifications required to become a web developer, a numerate degree in a subject such as maths or science will be useful.

You should also ideally have an aptitude for - or experience of - elements such as:

* User experience (UX)
* User interface (UI)
* Visual design
* Coding languages including HTML and CSS
* Frontend web programing languages and skills such as JavaScript, Ajax and web animation techniques
* Backend web programing languages such as C# or Java, PHP and Ruby
* Design software like Photoshop and Illustrator and Sketch
* An understanding of SEO
* Web servers and how they function

A career as a web developer is very specialised. While programs such as Adobe Dreamweaver and platforms such as WordPress are often perceived to take the place of core coding skills - and they do indeed allow novice users to create sites with basic knowledge - developing sites for corporate clients requires high levels of customisation which cannot be met with the use of programs such as Adobe Dreamweaver, as web developers are required to code in the raw language of the web.

A portfolio of your work is an ideal way to demonstrate your skills as a developer. Aim to include:

1. Examples of websites you've worked on - this allows you to share the work you have completed and helps to show prospective clients what you can do
2. Testimonials from clients you have previously worked with - this will reassure prospective clients that you have completed work for other clients and they were happy with what you delivered. Always ask any clients you work for to provide a testimonial once you complete a project
3. Your USPs - this is the ideal way to really sell yourself to prospective clients, showing what makes you stand out and why they should choose you over other developers
4. Your contact details - your name, email address, telephone number and social media handles, plus a link to your portfolio

Resources you may find helpful in building your portfolio include:

* [Sitepoint](https://www.sitepoint.com/how-to-create-a-portfolio-site-that-will-get-you-hired/)- tips on how to create a portfolio site to get you hired
* [Codementor](https://www.codementor.io/learn-programming/12-important-things-to-include-in-web-dev-portfolios) - advice on elements you should always include in your portfolio

If you are considering which subjects may stand you in good stead for a career in web development, consider numeracy-based subjects such as maths or science, plus subjects such as computer science.

## What skills do you need to become a web developer?

Key skills to be successful in web development include:

* Computer literacy
* Strong numeracy skills
* Strong creative ability
* Attention to detail
* Strong communication skills
* Excellent problem-solving skills
* A logical approach to work
* The ability to explain technical matters clearly
* A keen interest in technology

Ongoing self-learning is key to developing in a web development role, in order to stay up to date with ever more frequent technological advancements and updates.

* ***what is the role of a web developer***

A web developer builds and maintains websites with the client and consumer in mind. That is, the end design must include products and services offered and show how users may access these. For example, a customer may want a form to capture an end user’s e-mail to request additional information, provide a newsletter, or thank a customer for their business.

One type of developer may focus on setting up the back end of a site (back-end developer) while another may focus on the client side to add style and functionality to the website itself (front-end developer).

The work is typically very project-focused and involves collaborating with a team of people who help coordinate the client's needs with the end product. Work may include meeting with clients to discuss their needs and requirements for a website or discuss how to keep their website functioning and up-to-date. Web developers typically construct the layout of a website, creating a visually interesting home page and user-friendly design, and may sometimes write content for the website. After a website is up and running, developers make sure that the site is functional on all web browsers, testing and updating as needed.

Web developers are familiar with technology and understand how computers and web servers operate. They are also very familiar with many software programs, web applications and web programming languages, such as hypertext markup language (HTML), JavaScript (JS), Ruby on Rails, and C++. They must be able to communicate effectively, set goals and meet deadlines.