**Browser Timeline**

There’s an entire [history of web browsers](http://www.telegraph.co.uk/technology/microsoft/11577364/Web-browsers-a-brief-history.html). Before the web browsers we knew today, there were the first browsers, which are no longer in use or have highly evolved.

* **1990** – The WorldWideWeb (not to be confused with the World Wide Web) was the first browser ever created by W3C Director Tim Berners-Lee, then renamed Nexus to differentiate from the actual World Wide Web. Unlike today, this was the only browser and the only way to access the web.
* **1992** – Lynx was a texted-based browser that couldn’t display any graphic content.
* **1993**– Mosaic was the first browser to allow images embedded in text making it “the world’s first most popular browser”.
* **1994** – A noticeable improvement to Mosaic came Netscape Navigator.
* **1995** – Internet Explorer made its debut as Microsoft’s first web browser.
* **1996** – Opera started as a research project in 1994 that finally went public two years later. This was also arguably the beginning of the browser wars, mainly between IE 3 and Navigator 3 as Internet Explorer inched ahead with new capabilities.
* **2003** – Apple’s Safari browser was released specifically for Macintosh computers instead of Navigator.
* **2004** – Mozilla launched Firefox as Netscape Navigator faded out.
* **2007** – Mobile Safari was introduced as Apple’s mobile web browser and continues to dominate the iOS market.
* **2008** – Google Chrome appeared to soon take over the browser market.
* **2011** – Opera Mini was released to focus on the fast-growing mobile browser market.
* **2015** – Microsoft Edge was born to combat Google.

**What’s Behind a Browser?**

There are a few main components that make up the browser. For more of a deep dive to understand what happens between A (you clicking a link or typing in a URL) and B (you seeing a fully rendered web page), we can look closer at these elements to understand [how browsers work](https://www.html5rocks.com/tutorials/internals/howbrowserswork/#The_rendering_engine).

The UI, or user interface, is basically the browser wrapper. It includes the address bar, back and forward buttons, minimize/maximize/exit buttons, tabs, and the reload page button. These are standards in all browsers. Although they aren’t technically required, all these elements are usually found on every browser page to ensure usability.

The browser acts as a client to contact the web server and request information. When you search something on Google and click on a link or enter a URL in the address bar, the web server locates and sends the information you just requested to the web browser.

It does this by first performing a Domain Name System (DNS) look to find the correct IP address. This basically means that every URL has an equivalent set of numbers – the IP address – and the DNS translates it into language so we can type “Google.com” instead of “74.125.134.102”. After that, it sends a Hypertext Transfer Protocol (HTTP) request to the web server, which sends a response with HTML, CSS, and other content files to the browser.

The browser then communicates with the network to ask for all the documents that make up the page. Browsers can also display other more advanced material like interactivity and animations made with JavaScript with internal JavaScript interpreters.

The browser engine bridges the UI and the rendering engine. This is where the rendering engine reads the HTML and XML documents and interprets it to make a Document Object Model (DOM) tree and display the content. The rendering engine of every browser is different, which means a web page will likely look different depending on the browser.

**W3C Standards**

So why do browsers act differently? Why can you write a web application that works properly on Firefox, but when you pull it up on Safari, certain elements are hidden or unusable?

The World Wide Web Consortium is the standards organization for the web, which has a set of guidelines for browsers to adhere certain to HTML and DOM specifications.

Because it’s not a strict set of rules, they can be interpreted differently by different rendering engines. This is largely why you see discrepancies in browser experiences. Browsers can conform the specification while still abiding by their own rules, which can cause compatibility issues.

As browser versions update, many will add in other areas or features not specified in the W3C. Whether this be for competitive advantage or just because of a grey area in the guidelines, it can create cross-browser issues for software teams.

As we already know, the rendering engine is responsible for interpreting and displaying content. Since the [rendering engines of browsers are different](https://medium.com/@acparas/browsers-rendering-engines-js-engines-bea42b77a182), the content behaves different ways. Sometimes these differences are slight, while sometimes they’re major and can even make a web page unusable.

For example, Safari uses WebKit. Meanwhile, Chrome and Opera are both using Blink. Firefox uses Gecko and IE are using Trident. This means that each of these browsers abide by their own rules when it comes to rendering and displaying a web page, which can be a huge pain for developers trying to create a consistent experience.

**Browser Wars**

As mentioned, [browser wars](https://en.wikipedia.org/wiki/Browser_wars) have been happening since Internet Explorer and Netscape Navigator went up against each other to be best in class browser of the 90’s. Netscape had previously been leading the market, but with the release of Internet Explorer 3, Microsoft took the lead. Since it was automatically included in the Windows OS, it became the standard for many desktop users.

The same situation happened when Safari came to be in 2003. Whereas Macintosh users were previously on IE or Navigator, Safari being preinstalled in Apple’s OS mean that it gained control of that desktop market.

After some time of IE beating out Navigator, Netscape made the code for the browser open source and gave it to Mozilla. However, when Firefox was introduced in 2004, it saw a rapid rise in popularity for a few years. That is until Google Chrome was released in 2008 and quickly fought to be the favored browser, which we still see today.

Of course, in between, we’ve also seen increased mobile use and differences in mobile browsers as well as the introduction of HTML5 and CSS3. This all contributes to the browser wars between Internet Explorer, Safari, Firefox, Chrome, Opera, and, recently, Edge to be the preferred browser for surfing the World Wide Web.

That competition comes in the form of frequent browser updates and version releases that you may notice today as each browser tries to one-up the other in speed, security, features, and design. For developers, many will try to program in a popular browser or a browser with a standard rendering engine so that the page is cross-compatible. For example, most people avoid developing in Internet Explorer since it’s known to be so problematic.

**Modern Browsing**

Today, [Chrome still rules desktop browsers](https://www.w3schools.com/browsers/) while [Safari owns the mobile browsing market](https://www.statista.com/statistics/272664/market-share-held-by-mobile-browsers-in-the-us/). However, there still is no defined winner. The browser wars are still going strong and fragmentation is a more prevalent issue than ever due to the frequent updates and releases of different browser versions and operating systems.

Browsers preferences are also largely dependant on demographics including age, country, and even job. For example, many schools and companies have certain requirements about what devices, operating systems, and browsers users may access and don’t permit individuals to update on their own.

Additionally, as browsers continually update versions hoping to be the next Google Chrome, there’s no saying when one will finally surpass its popularity or when another company creates their own browser to enter the mix. In fact, with [Firefox’s recent Quantum version](https://crossbrowsertesting.com/blog/test-automation/mozilla-firefox-quantum-browser/), more and more users and considering making the switch in favor of a faster browsing experience.

Not to mention, the underdog browsers still capture a distinct user base. While something like UC Browser might not capture a large percentage of the internet in comparison to the big five, it still serves about [500 million people](https://en.softonic.com/articles/uc-browser-reaches-over-500-million-users-worldwide).

So, what lessons can we take away from the history of web browsers? As software professionals, the only sure-fire way to approach this diverse user behavior is to develop web applications to be cross-compatible and test them across different browsers to make sure they’re visually and functionally acceptable.

Now that you know browser history,