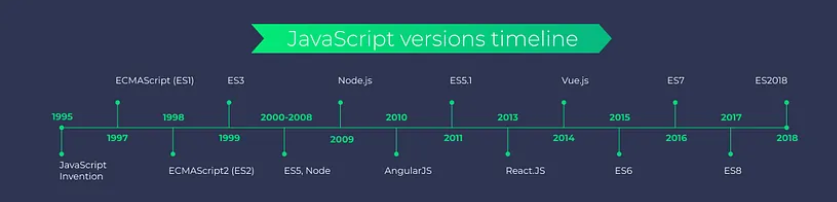
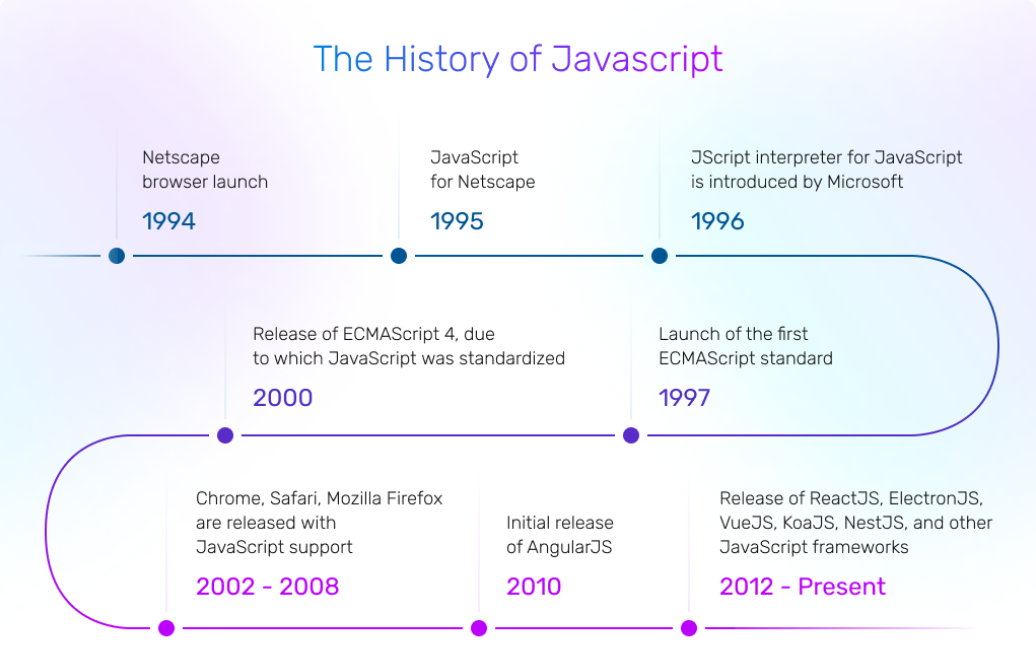
**Topic – 1 : History of JavaScript**

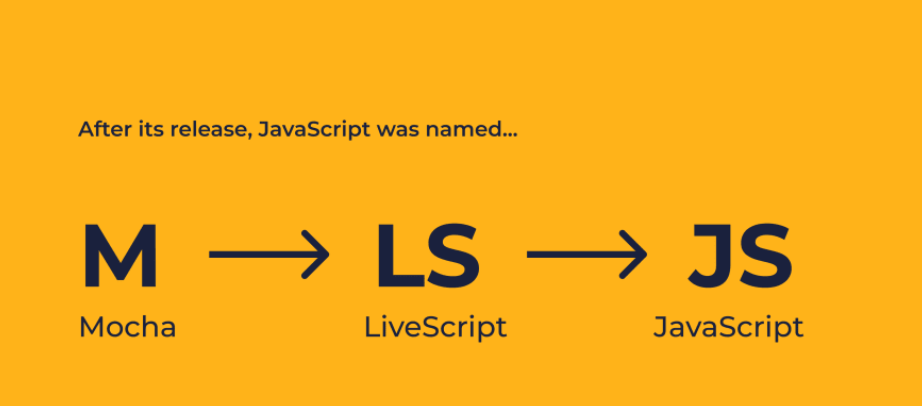
Creation: **Brendan Eich** created JavaScript in 1995 while working at Netscape Communications Corporation. It was originally called Mocha, then changed to LiveScript, before finally becoming JavaScript.

* Implementation: JavaScript was first implemented in the Netscape Navigator 2.0 browser.
* ECMA-262 standard: In 1997, JavaScript became the ECMA-262 standard.
  + 
* Mozilla: After Netscape handed JavaScript over to ECMA, the Mozilla foundation continued to develop JavaScript for the Firefox browser.
* Purpose: JavaScript was designed to add interactivity to web pages and enable developers to create dynamic content.
* Popularity: JavaScript was named after Java, the most popular programming language at the time.
* Uses: JavaScript is used for many purposes, including building mobile applications, developing the back end infrastructure of web applications, and developing browser based games.



**Topic – 2 : How JavaScript named as JavaScript?**

In September 1995, a **Netscape programmer named Brendan Eich** developed a new scripting language in just 10 days. It was originally called Mocha, but quickly became known as LiveScript and later JavaScript.



The language derived its syntax from Java, its first-class functions from scheme, and its prototype-based inheritance from self.  
Since then, JavaScript has been adopted by all major graphical web browsers.

**Topic – 3 : What is relation with Java ?**

JavaScript and Java are two distinct programming languages, despite sharing similar names.

* Their relationship is mostly historical and superficial.
* JavaScript and Java are not directly related in terms of functionality or design, and the
* similarity in their names is largely coincidental due to marketing.
* They serve different purposes and have different language designs, but both have played
* significant roles in their respective fields.

**Topic – 4 : Is JavaScript only used for Frontend?**

JavaScript is not only a front-end language, and web development is not the only type of front-end development, and it all depends on how you define “programming language”, but at time of writing, yes, procedural code that runs in web browsers is JavaScript 99.9% of the time. The remaining one thousandth is either old browsers running flash, silverlight, or java applets, or code compiled from another language into the web assembly binary format.

Javascript is the predominant language for writing scripts on the front end of web applications.

However, it's not the only language used for this purpose. There are other languages that compile or transpile to JavaScript, offering different approaches to frontend development:

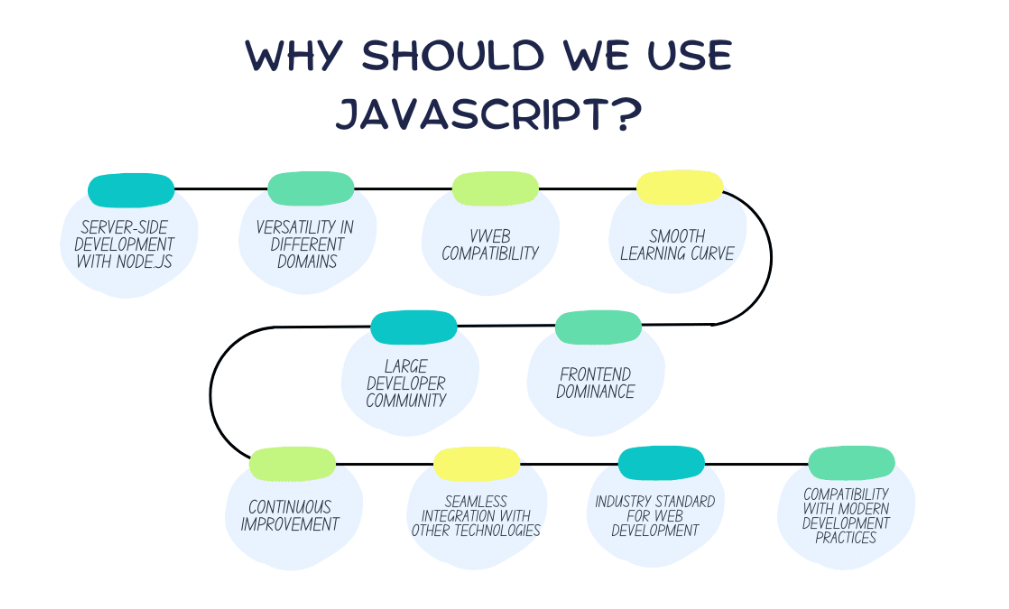
1. **TypeScript:** Developed by Microsoft, TypeScript is a superset of JavaScript that adds static typing to the language. It compiles down to JavaScript and is widely used in large-scale applications due to its added safety and maintainability.
2. **CoffeeScript:** It's a language that compiles into JavaScript, offering a more concise and expressive syntax. While its popularity has decreased over time, some developers still use it for its brevity.
3. **Dart:** Originally created by Google, Dart is a language with its own runtime that can be compiled to JavaScript. It's commonly used with the Flutter framework for building cross-platform mobile, web, and desktop applications.
4. **Elm:** Elm is a functional programming language that compiles to JavaScript and is specifically designed for building frontend applications. It focuses on simplicity, maintainability, and reliability.
5. **ClojureScript:** Based on Clojure, a dialect of Lisp, ClojureScript compiles to JavaScript and emphasizes functional programming. It's used by developers who appreciate Lisp's expressive power and functional programming features.

These languages offer different paradigms, syntaxes, and features while**ultimately targeting JavaScript as the language executed in the browser.** JavaScript, however, remains the primary language for frontend web development due to its native support in web browsers.

**Topic – 5 : Why and What JavaScript?**

**Why JavaScript?**

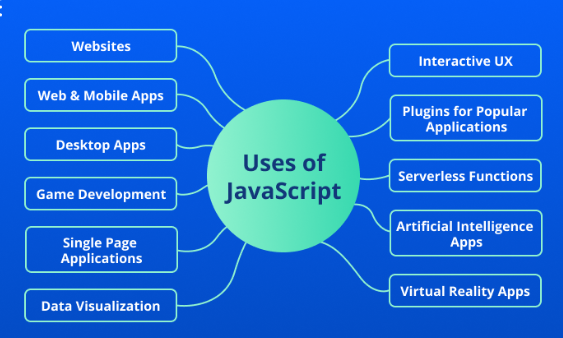
JavaScript is a popular programming language for a number of reasons, including:



* **Easy to learn:** JavaScript uses a standard syntax that's similar to other programming languages, such as C, Java, and Python.
* **Browser support:** All browsers support JavaScript, which allows for the display of videos, animations, and other media.
* **Versatility:** JavaScript can be used on both the front and back end of websites.
* **Powerful frameworks:** JavaScript has many frameworks that provide ready-to-use codes that are easy to understand and troubleshoot.
* **Less server interaction:** JavaScript allows for the validation of user input before sending the page to the server, providing immediate feedback to the user.
* **Compatibility with other languages:** JavaScript is compatible with other languages, which is important because web servers can run on different languages.
* **Part of the front-end programming language triad:** JavaScript is one of the three front-end programming languages, along with CSS and HTML.

**What JavaScript?**

JavaScript is a scripting language that's used to add complex features to web pages and mobile apps:



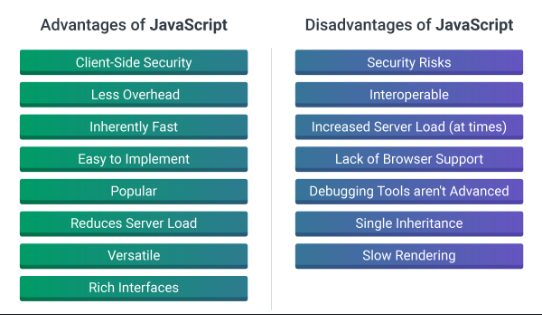
* **Web pages:** JavaScript makes web pages more interactive and dynamic. It can also allow users to load content into a document without reloading the entire page.
* **Mobile apps:** JavaScript frameworks like React Native, Ionic, NativeScript, and Apache Cordova can be used to build native and hybrid apps for Android and iOS.
* **Games:** JavaScript can be used to build games when combined with HTML5 and an Application Programming Interface (API) like WebGL. Some games made with JavaScript include Angry Birds, The Wizard, and 2048.

JavaScript is popular for a number of reasons, including:

* **Works in most browsers:** JavaScript is one of the few programming languages that can be used in all popular browsers.
* **Many frameworks and libraries:** There are many ready-to-go frameworks and libraries that can help speed up development.
* **Easy to find developers:** The popularity of JavaScript means that there are many JavaScript developers available.

JavaScript's syntax is based on the Java and C languages, and it supports both object-oriented and functional programming.

**Topic – 6 : Advantages and Dis-Advantages of JavaScript**



**Topic – 7 : Synchronous vs Asynchronous?**

In JavaScript, synchronous and asynchronous refer to how tasks are executed, and the main differences are:

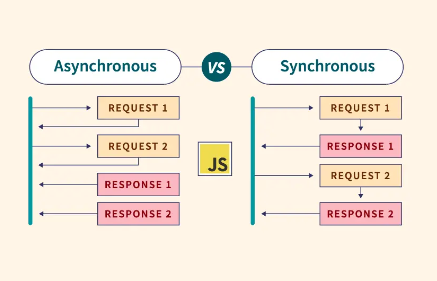
* **Synchronous**

Tasks are executed sequentially, one after the other, and the main thread waits for each task to complete before moving on. This can cause delays if a task takes a long time to complete. Synchronous code is easier to implement because it executes linearly without callback functions or promises. However, it can be difficult to scale and communicate between threads.

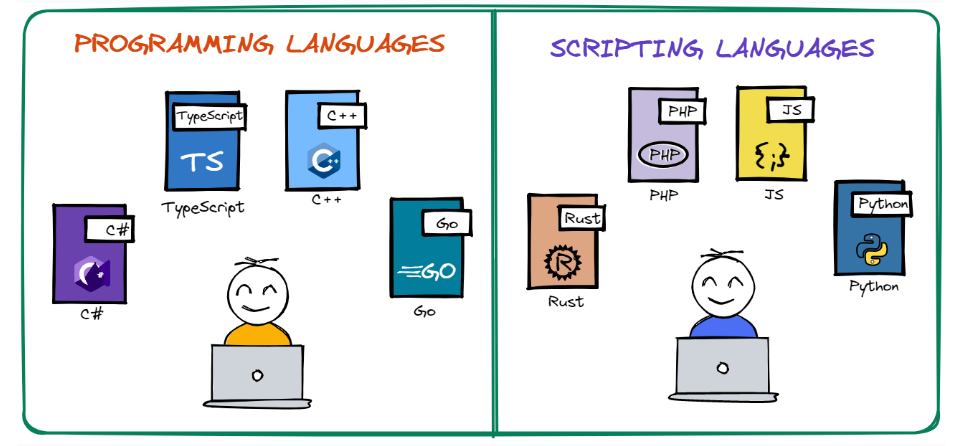
* **Asynchronous**

Tasks are executed non-sequentially, allowing other code to run simultaneously. Asynchronous code uses callbacks and Web APIs to delegate time-consuming tasks to the browser's background processes. This allows developers to continue executing code without waiting for tasks to finish. Asynchronous code is more efficient because multiple operations can run at the same time.

Understanding the differences between synchronous and asynchronous JavaScript is important for developers to write efficient and responsive code.

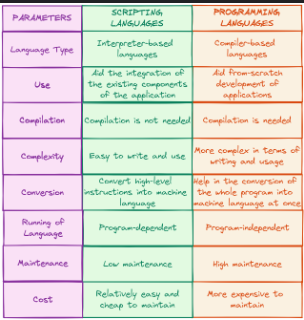


**Topic – 8 : Difference between scripting and programming language?**

****

Scripting and programming languages differ in several ways, including their purpose, execution, and compilation:

* **Purpose:** Programming languages are used to develop programs and communicate with computers. Scripting languages are used to build websites and web applications by linking together existing parts of a program.
* **Execution:** Programming languages are compiled, meaning they are translated into machine code before being executed. Scripting languages are interpreted, meaning they are executed directly by an interpreter without being compiled.
* **Flexibility:** Scripting languages are more dynamic and can be changed while running, while programming languages must be recompiled after changes.
* **Ease of use:** Scripting languages are designed to be flexible and easy to use, making them well-suited for rapid development and task automation.

****