What is the event loop?

In Node.js, the event loop is the mechanism that performs the non-blocking, asynchronous I/O operations. It listens for incoming events (such as incoming network connections, completed file I/O operations, timers, etc.) and executes the corresponding event handlers.

The event loop runs on a single thread, which is different from traditional web servers that create a new thread for each incoming request. Instead, Node.js uses an event-driven, callback-based model, where the event loop acts as the "coordinator" that manages the execution of the event handlers. This allows Node.js to handle a large number of connections with a small number of threads, making it well-suited for server-side applications that handle a high volume of network traffic.

What are the components of the event loop?

1.              Task Queue: This is where the callbacks for various events are stored, waiting to be executed by the event loop.

2.               Heap: This is where JavaScript objects and variables are stored in memory.

3.              Stack: This is where the JavaScript function calls are stored as they are executed.

4.              Tick: This refers to a single iteration of the event loop. During each tick, the event loop will check the task queue for pending callbacks and execute the next callback in the queue, if one is available.

5.              Poll: This is a phase of the event loop where the event loop checks the task queue for new events and executes any available callbacks.

6.              Check: This is a phase of the event loop where the event loop checks the task queue for low-priority events, such as setImmediate(), and executes the corresponding callbacks.

7.              Close Callbacks: This is a phase of the event loop where the event loop executes close callbacks, such as those associated with sockets and file streams that are being closed.

The event loop in Node.js continues to run until there are no more events to process or the process is terminated. The order in which the phases of the event loop are executed may vary depending on the implementation, but the overall structure remains the same.

Event Emitter:

The Event Emitter is a class in Node.js that allows objects to emit events and register listeners for those events. It provides a simple way to implement the observer pattern in Node.js, where objects can subscribe to events and get notified when those events occur.  Event Emitter can be used as a base class for custom objects that emit events, or it can be used directly.

To use Event Emitter, you create an instance of the class, then use the. on() method to register event listeners, and the .emit() method to trigger events. When an event is triggered, all listeners registered for that event are called in the order they were registered.  Event Emitter is widely used in Node.js and its modules, providing a flexible and powerful way to handle events and callbacks. It is especially useful for implementing custom events in applications and libraries, allowing objects to communicate with each other in a decoupled manner.