Intro to NodeJS

Héctor Vásquez
Junior Engineering Manager @ Soluciones GBH

Agenda

- NodeJS| What is it?
- NodeJS| Event Loop
    - Event Loop| Callbacks
    - Event Loop| Overview
    - Event Loop| Phases
- NodeJS| Q&A

NodeJS| What is it?

- An asynchronous event-based JavaScript runtime built on top of Google's V8

- Designed to build scalable network apps with the Event Loop at its core

- No direct I/O operations === process never blocks

Agenda

- NodeJS| What is it?
- NodeJS| Event Loop
  - Event Loop| Callbacks
  - Event Loop| Overview
  - Event Loop| Phases
- NodeJS| Q&A

| Event Loop| Callbacks

- Are functions called at the completion of a given task
- Prevent process blocking
- Allows other code to run in the meantime
- Are the foundation of NodeJS

```
function processData () {
  var data = fetchData ();
  data += 1;
  return data;
}
```

```
function processData (callback) {
  fetchData(function (err, data) {
    if (err) {
      console.log("An error has occurred. Abort everything!");
      return callback(err);
    }
    data += 1;
    callback(data);
  });
}
```

Agenda

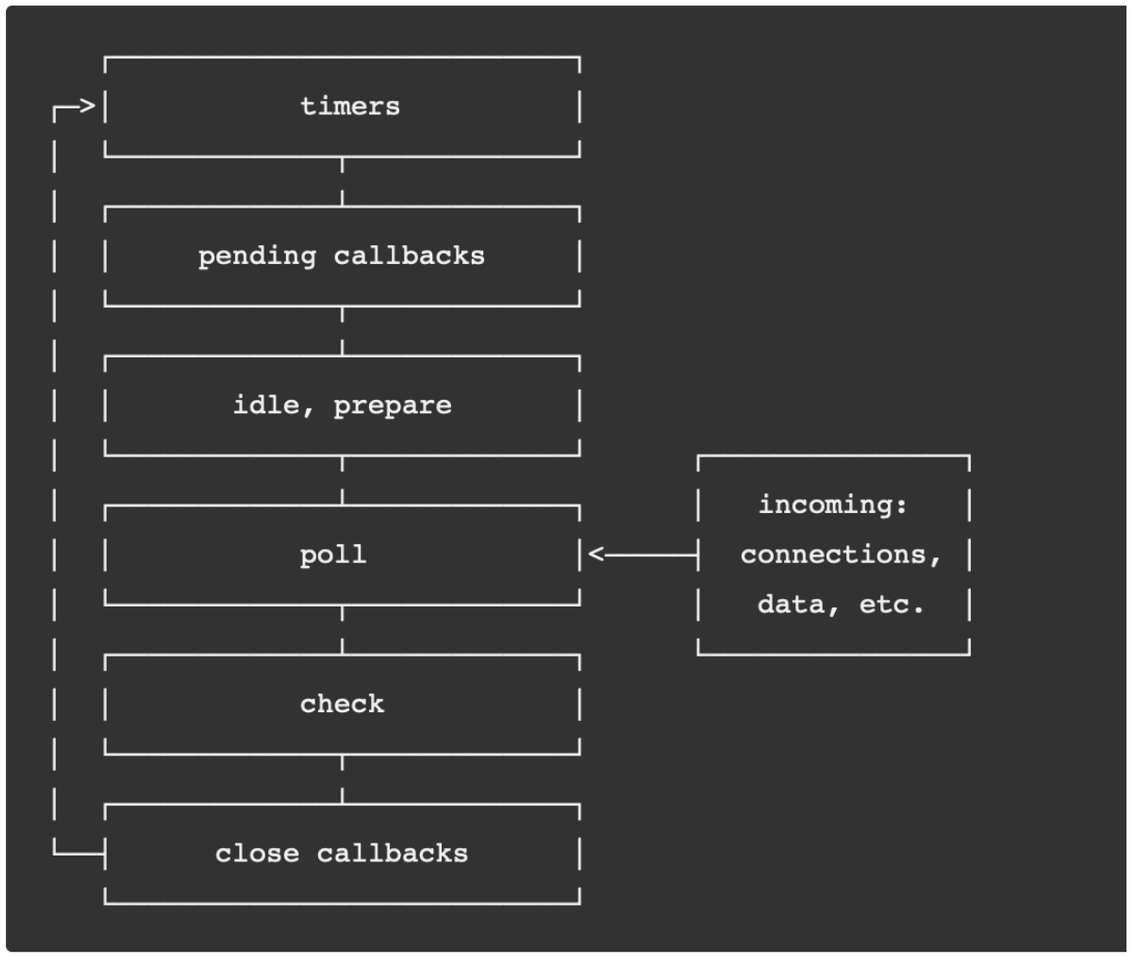
- NodeJS| What is it?
- NodeJS| Event Loop
    - Event Loop| Callbacks
    - Event Loop| Overview
    - Event Loop| Phases
- NodeJS| Q&A

| Event Loop| Overview

- Allows NodeJS to perform non-blocking I/O operations by offloading operations to the system kernel

- It is composed by phases, each with a FIFO queue of callbacks to execute

- Each phase executes phase-specific operations and certain amount of callbacks before moving to the next phase



note: each box will be referred to as a "phase" of the event loop.

Agenda

- NodeJS| What is it?
- NodeJS| Event Loop
    - Event Loop| Callbacks
    - Event Loop| Overview
    - Event Loop| Phases
- NodeJS| Q&A

| Event Loop| Phases (1)

- **Timers**
    - Executes callbacks scheduled by `setTimeout()` and `setInterval()`
- **Pending Callbacks**
    - Executes I/O callbacks deferred to the next loop iteration
- **Idle, Prepare**
    - Only used internally

| Event Loop| Phases (2)

- **Poll**
    - Calculates how long should block and retrieve new events
    - Retrieves new I/O events and executes I/O related callbacks
    - If no scheduled timers (e.g. `setTimeout()`) and
        - Poll queue is not empty, then iterates through queue until it is empty or execution limit is hit
        - Poll queue is empty and
            - Callbacks scheduled with `setImmediate()`, then moves to next phase
            - No callbacks scheduled with `setImmediate()`, then waits for callbacks to be added to the queue and executes them immediately

| Event Loop| Phases (3)

- **Check**
  - Executes callbacks scheduled with `setImmediate()`
- **Close Callbacks**
  - Executes close callbacks (e.g. `socket.on('close', ...)`)

Agenda

- NodeJS| What is it?
- NodeJS| Event Loop
    - Event Loop| Callbacks
    - Event Loop| Overview
    - Event Loop| Phases
- NodeJS| Q&A

NodeJS| Q&A