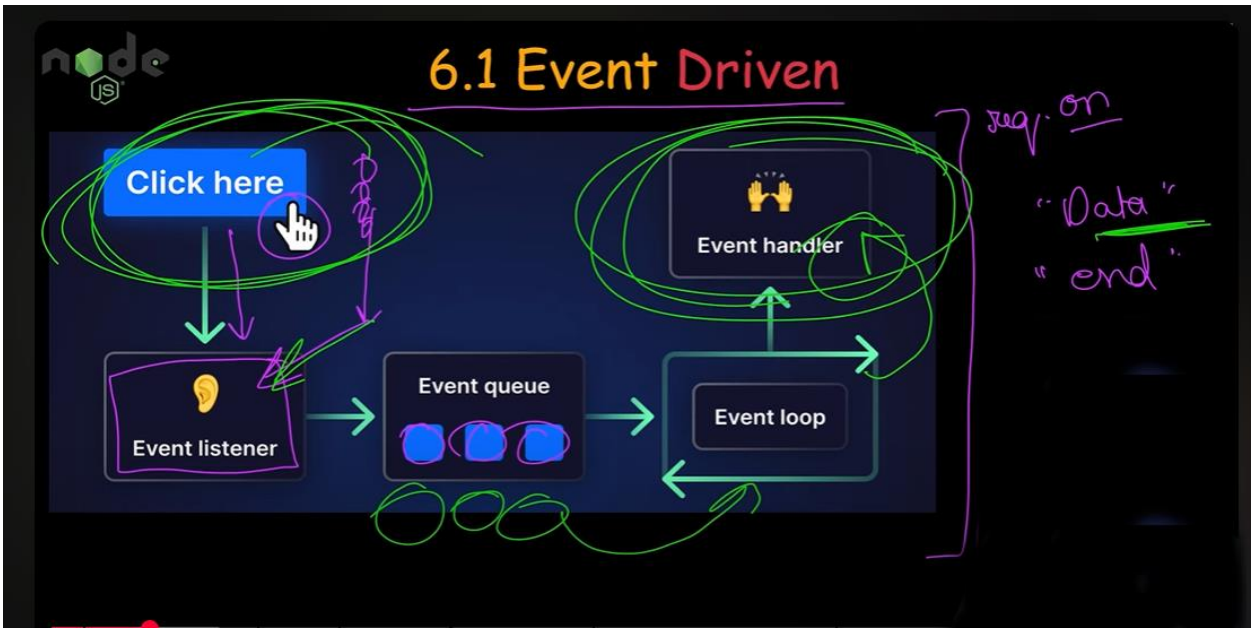
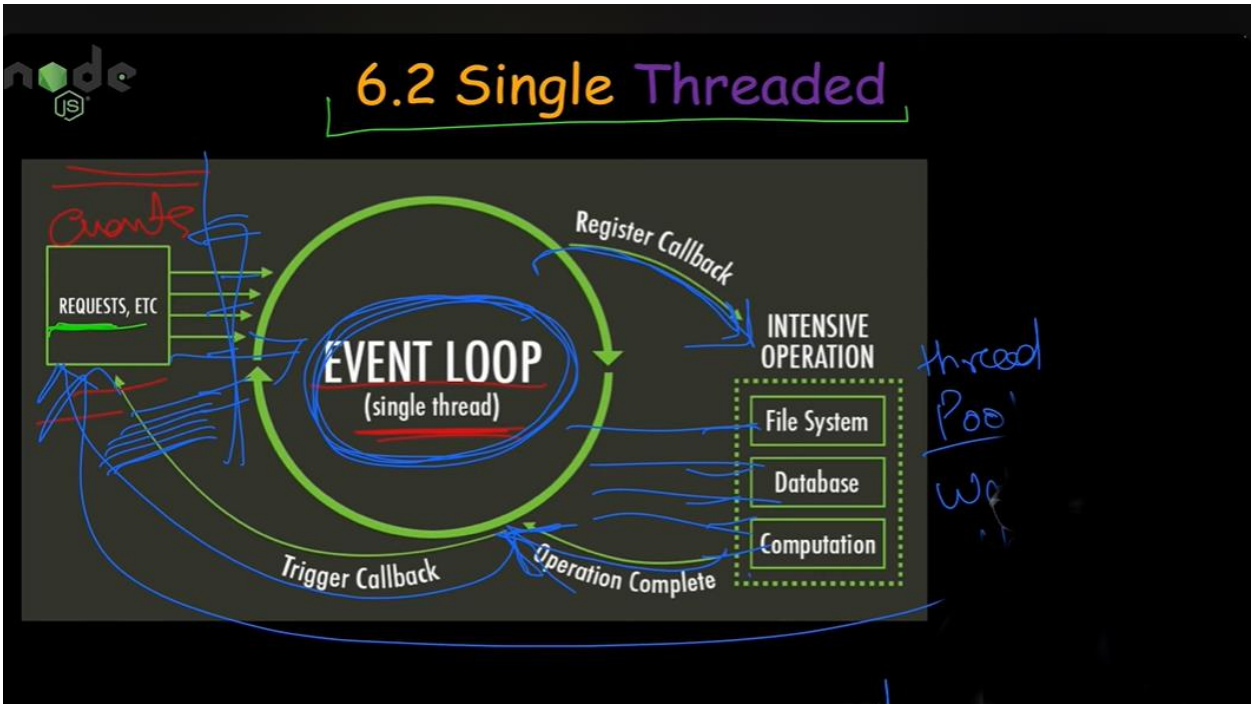
Event Driven and Async Code::





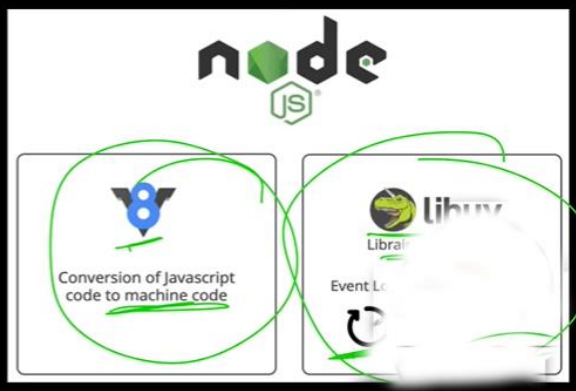
6.3 V8 vs libuv

V8:

1. Open-source JavaScript engine by Google.
2. Used in Chrome and Node.js.
3. Compiles JavaScript to native machine code.
4. Ensures high-performance JavaScript execution.

libuv:

1. Multi-platform support library for Node.js.
2. Handles asynchronous I/O operations.
3. Provides event-driven architecture.
4. Manages file system, networking, and timers non-blockingly across platforms.



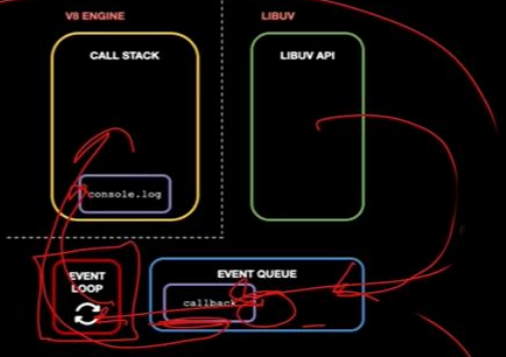
6.4 Node Runtime

Callbacks of completed queries are moved to the event queue. If the call stack is empty, the event loop checks for callbacks and transfers the first.

```
console.log("Starting Node.js");

db.query("SELECT * FROM public.cars", function (err, res) {
  console.log("Query executed");
});

console.log("Before query result");
```

OUTPUT

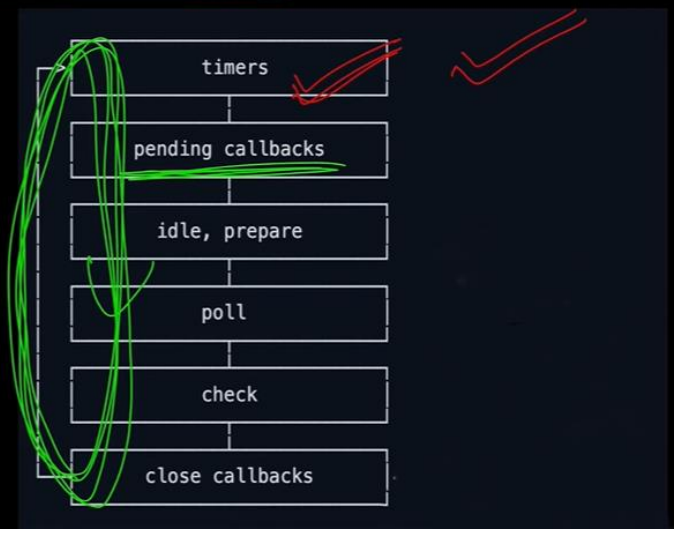
```
Starting Node.js
Before query result
```



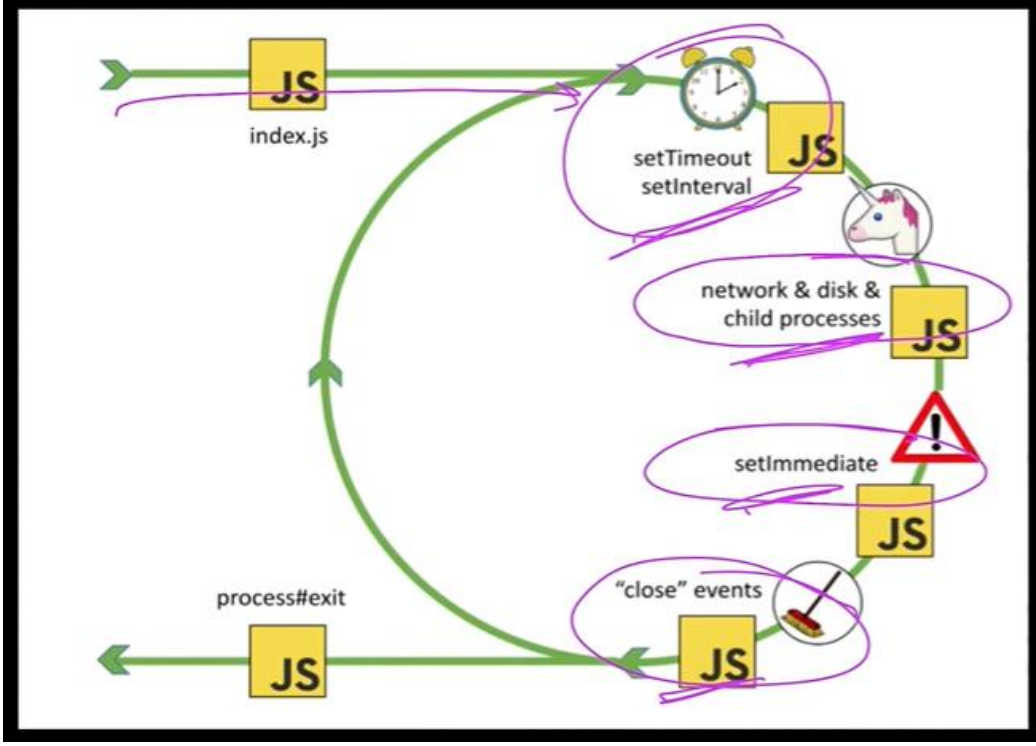
Made with ❤️ by @FabriLallo and @AndrewHu368

6.5 Event Loop

- timers: this phase executes callbacks scheduled by setTimeout() and setInterval().
- pending callbacks: executes I/O callbacks deferred to the next loop iteration.
- idle, prepare: only used internally.
- poll: retrieve new I/O events; execute I/O related callbacks (almost all with the exception of close callbacks, the ones scheduled by timers, and setImmediate()); node will block here when appropriate.
- check: setImmediate() callbacks are invoked here.
- close callbacks: some close callbacks, e.g. socket.on('close', ...).



6.5 Event Loop



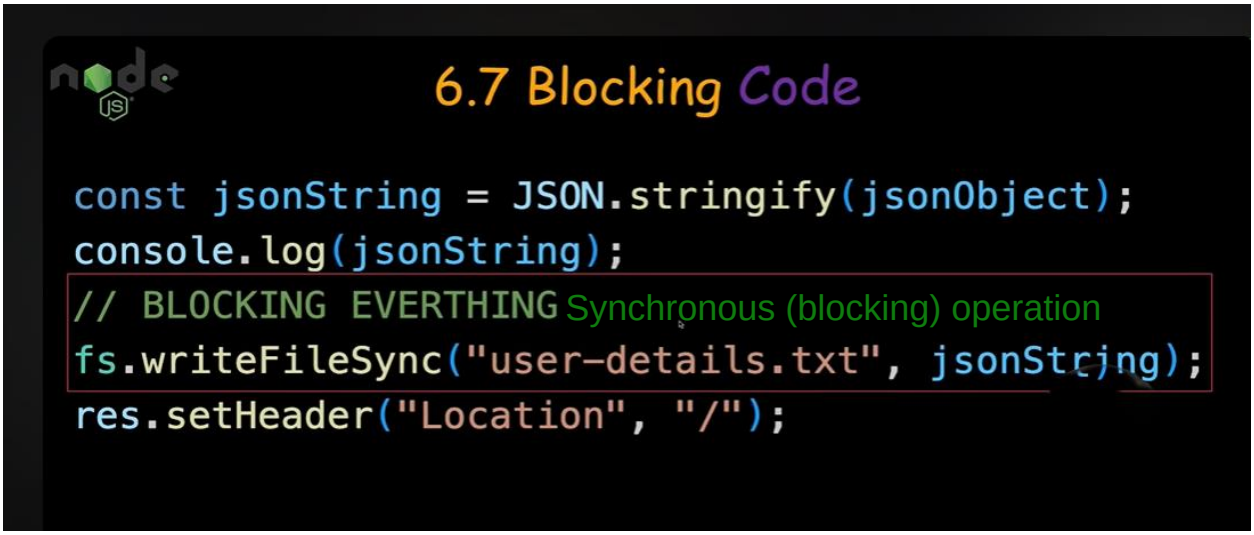
6.6 Async Code

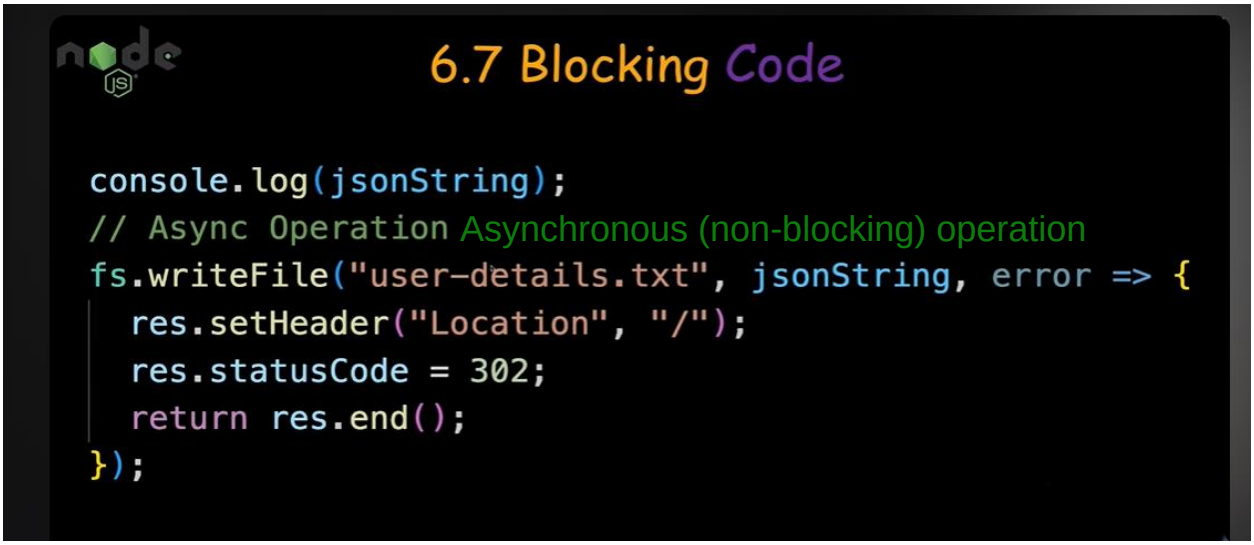
```
    const jsonString = JSON.stringify(jsonObject);
    console.log(jsonString);
    fs.writeFileSync("user-details.txt", jsonString);
    res.setHeader("Location", "/");
    res.statusCode = 302;
    res.end();
  });
}

res.write('<body><h1>Like / Share / Subscribe</h1></
body>');
res.write('</html>');
return res.end();
```

```
Error [ERR_HTTP_HEADERS_SENT]: Cannot set headers after they are sent to the client
    at ServerResponse.setHeader (node:_http_outgoing:699:11)
    at IncomingMessage.<anonymous> (/Users/prashantjain/workspace/Test Project/node/app.js:44:11)
    at IncomingMessage.emit (node:events:532:35)
    at endReadableNT (node:internal/streams/readable:1696:12)
    at process.processTicksAndRejections (node:internal/process/task_queues:82:21) {
  code: 'ERR_HTTP_HEADERS_SENT'
}
```

6.6 Async Code

```
req.on("end", () => {
  const parsedBody = Buffer.concat(body).toString();
  console.log(parsedBody);
  const params = new URLSearchParams(parsedBody);
  const jsonObject = {};
  for (const [key, value] of params.entries()) {
    jsonObject[key] = value;
  }
  const jsonString = JSON.stringify(jsonObject);
  console.log(jsonString);
  fs.writeFileSync("user-details.txt", jsonString);
  res.setHeader("Location", "/");
  res.statusCode = 302;
  return res.end();
});
```


6.7 Blocking Code
const jsonString = JSON.stringify(jsonObject);
console.log(jsonString);
// BLOCKING EVERTHING Synchronous (blocking) operation
fs.writeFileSync("user-details.txt", jsonString);
res.setHeader("Location", "/");


6.7 Blocking Code
console.log(jsonString);
// Async Operation Asynchronous (non-blocking) operation
fs.writeFile("user-details.txt", jsonString, error => {
res.setHeader("Location", "/");
res.statusCode = 302;
return res.end();
});