

Managing HTTP Responses



Armen Avanesi
SOFTWARE DEVELOPER

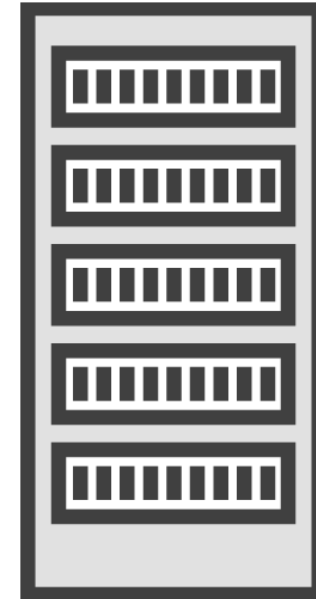




request



response



Creating a Response

Set status code

Set headers

Write data

Close connection



Challenge

Set the “Content-Type” header to
“application/json”



Responding with JSON



The Benefits of Streamed Data



What Benefits Do Streams Actually Provide Us?



Benefits

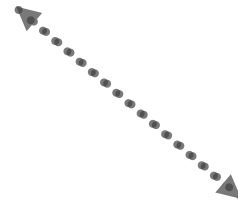
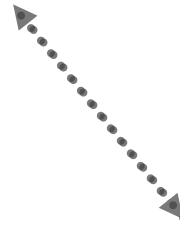
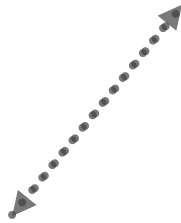
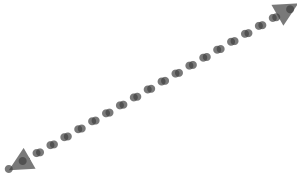
Memory Efficiency

Time Efficiency

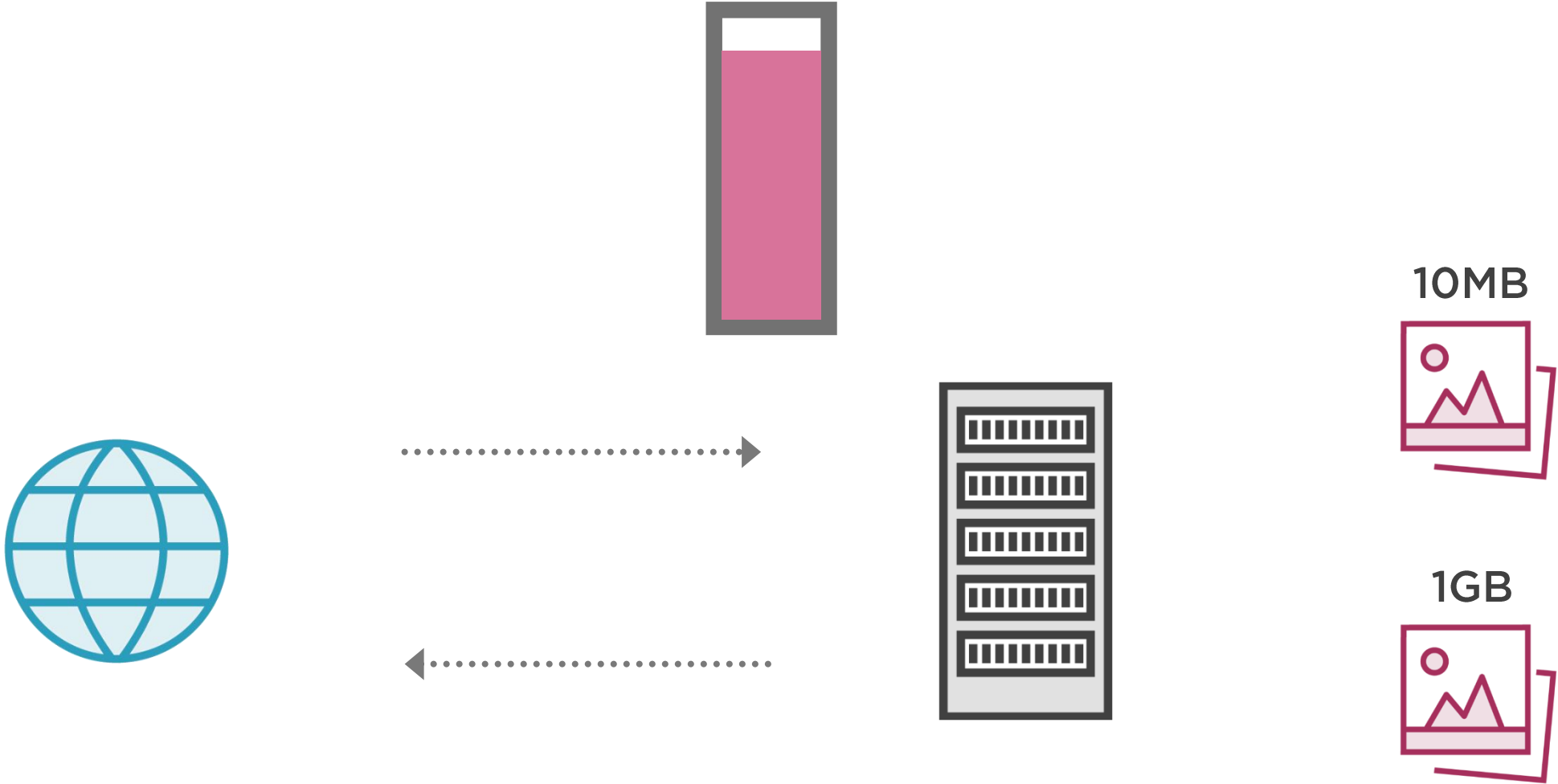


We can operate on our data
one piece at a time

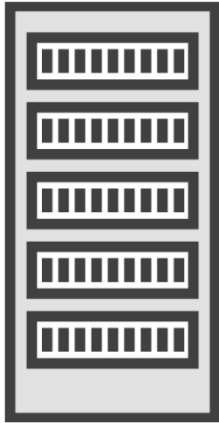
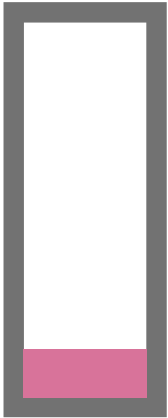




Server Memory



Server Memory

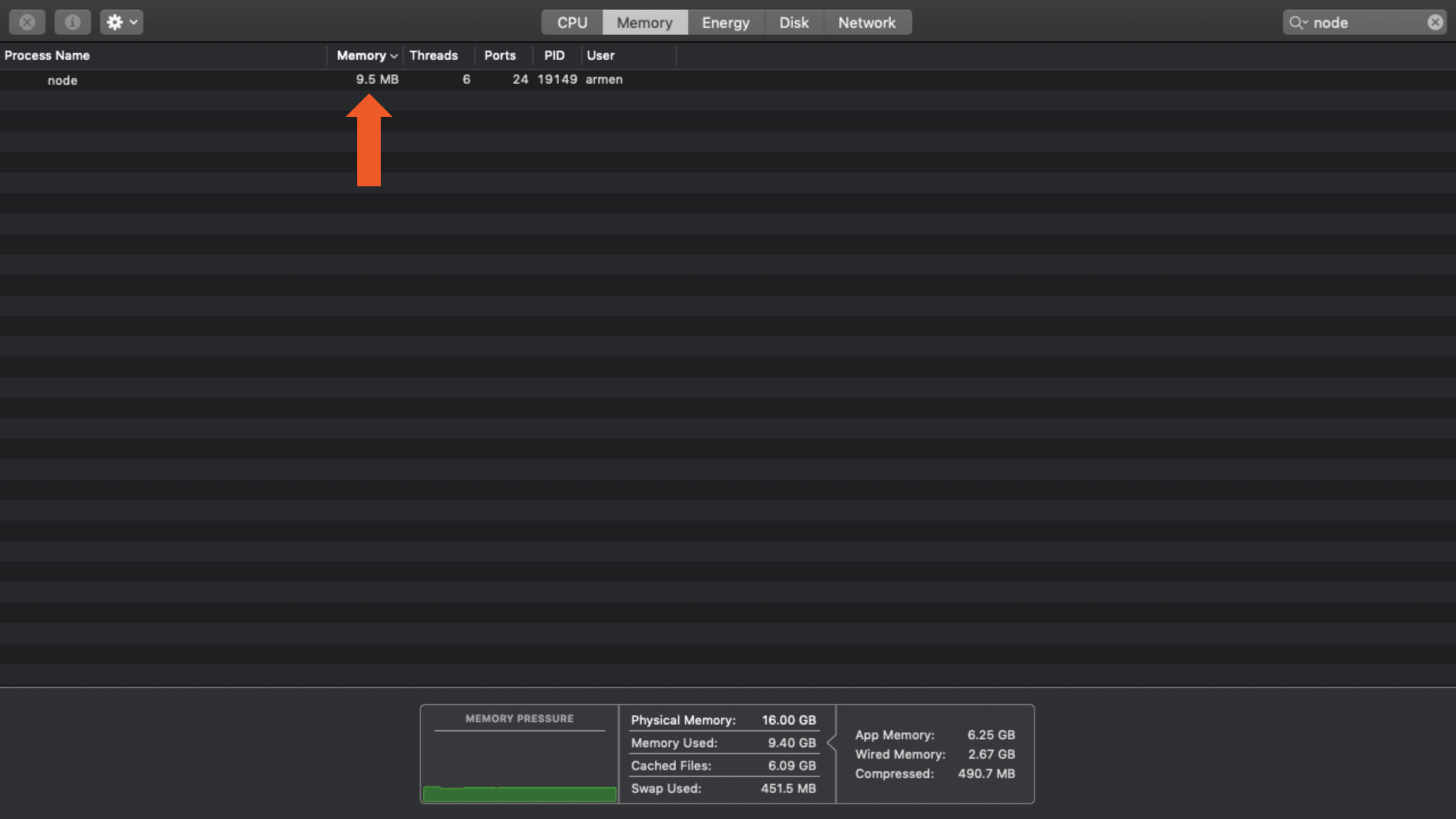


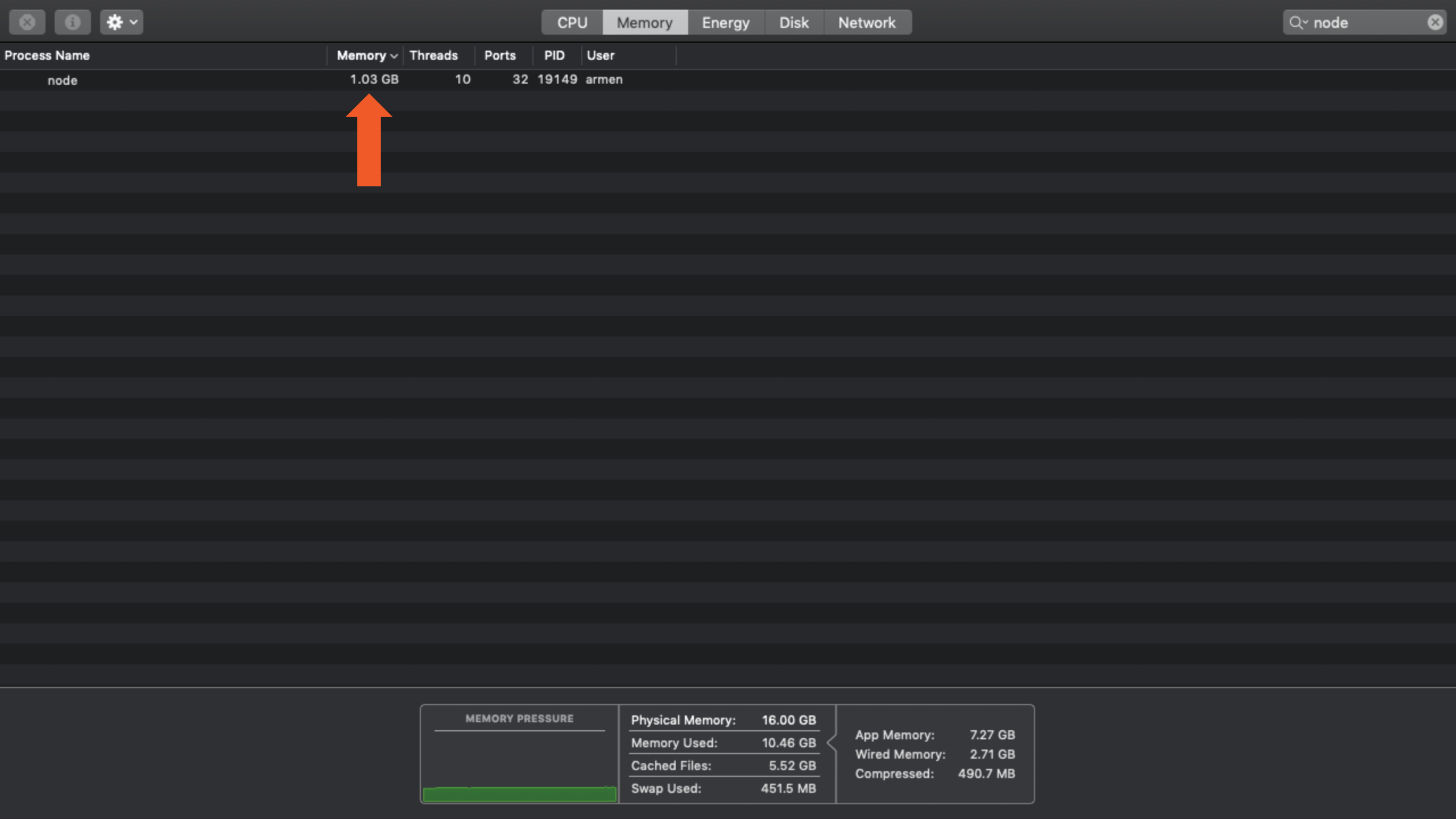
10MB

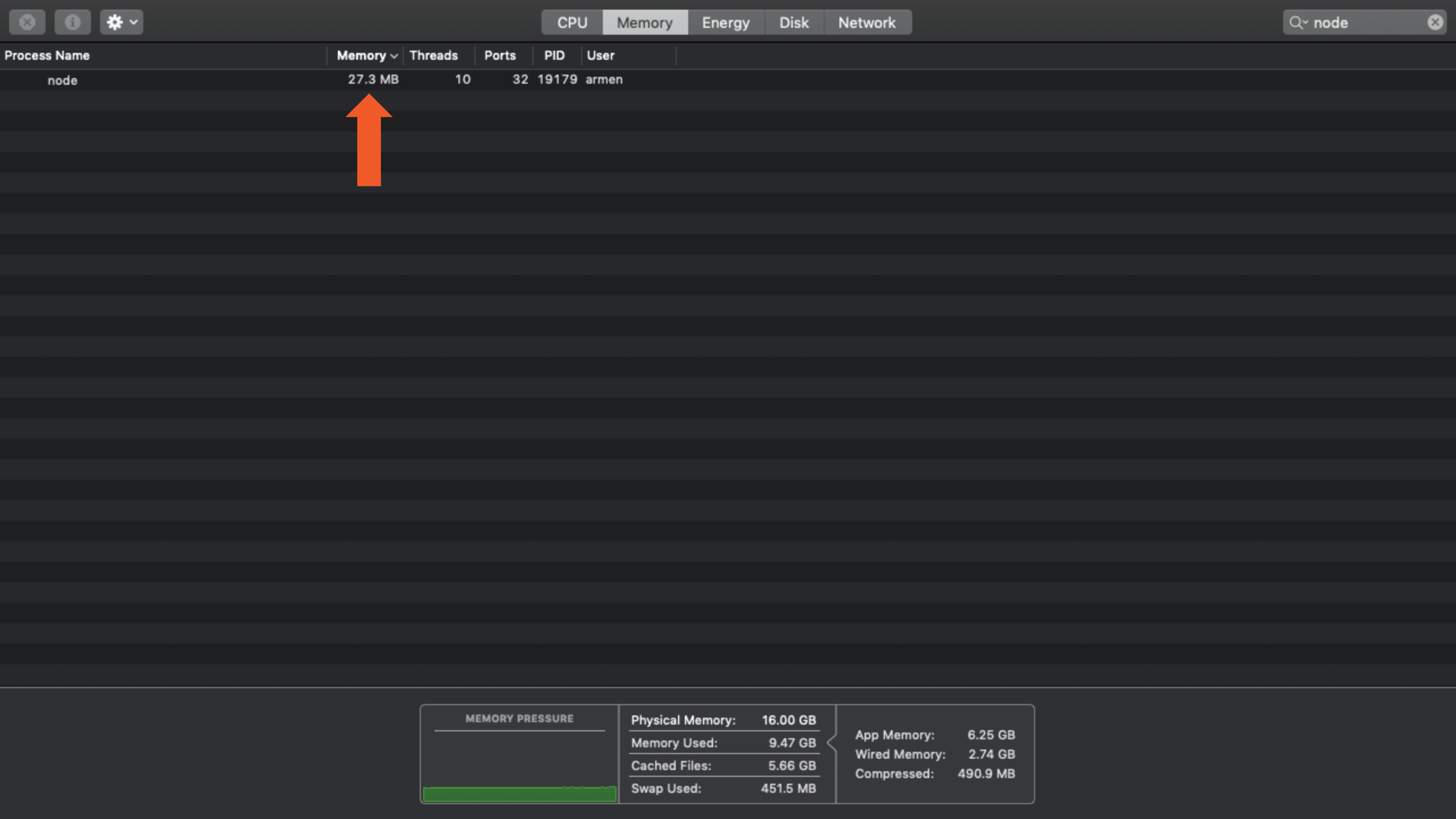


1GB







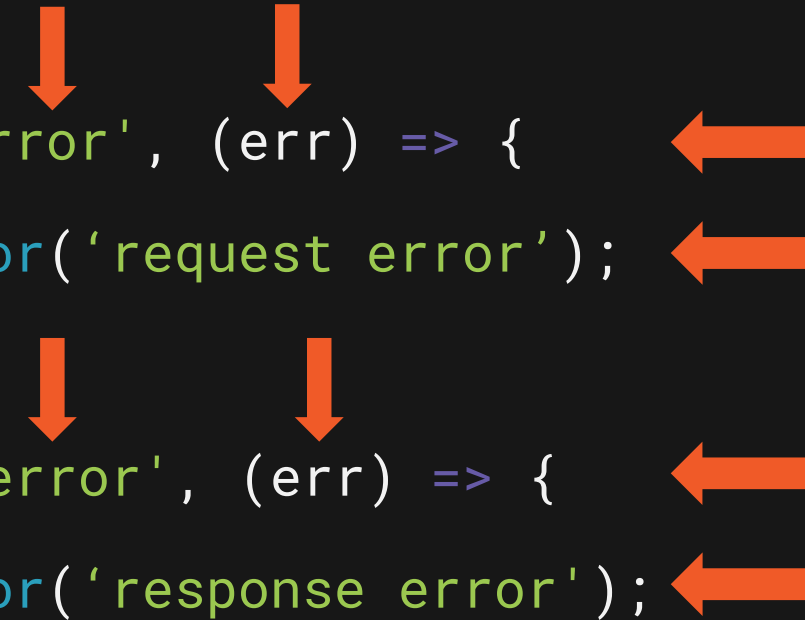


Handling Errors



server.js

```
request.on('error', (err) => {  
    console.error('request error');  
});  
response.on('error', (err) => {  
    console.error('response error');  
});
```



The “Error” Class

`error.code`

`error.message`

`error.stack`



server.js

```
response.statusCode = 500;
```

```
response.write("An error has occurred");
```

```
response.end();
```

Performance Hooks

Policies

Process

Punycode

Query Strings

Readline

REPL

Report

Stream

String Decoder

Timers

TLS/SSL

Trace Events

TTY

UDP/Datagram

URL

Utilities

V8

VM

WASI

Worker Threads

Zlib

- **EACCES** (Permission denied): An attempt was made to access a file in a way forbidden by its file access permissions.
- **EADDRINUSE** (Address already in use): An attempt to bind a server (`net` , `http` , or `https`) to a local address failed due to another server on the local system already occupying that address.
- **ECONNREFUSED** (Connection refused): No connection could be made because the target machine actively refused it. This usually results from trying to connect to a service that is inactive on the foreign host.
- **ECONNRESET** (Connection reset by peer): A connection was forcibly closed by a peer. This normally results from a loss of the connection on the remote socket due to a timeout or reboot. Commonly encountered via the `http` and `net` modules.
- **EEXIST** (File exists): An existing file was the target of an operation that required that the target not exist.
- **EISDIR** (Is a directory): An operation expected a file, but the given pathname was a directory.
- **EMFILE** (Too many open files in system): Maximum number of `file descriptors` allowable on the system has been reached, and requests for another descriptor cannot be fulfilled until at least one has been closed. This is encountered when opening many files at once in parallel, especially on systems (in particular, macOS) where there is a low file descriptor limit for processes. To remedy a low limit, run `ulimit -n 2048` in the same shell that will run the Node.js process.
- **ENOENT** (No such file or directory): Commonly raised by `fs` operations to indicate that a component of the specified pathname does not exist — no entity (file or directory) could be found by the given path.
- **ENOTDIR** (Not a directory): A component of the given pathname existed, but was not a directory as expected. Commonly raised by `fs.readdir`.
- **ENOTEMPTY** (Directory not empty): A directory with entries was the target of an operation that requires an empty directory — usually `fs.unlink`.
- **ENOTFOUND** (DNS lookup failed): Indicates a DNS failure of either `EAI_NODATA` or `EAI_NONAME`. This is not a standard POSIX error.
- **EPERM** (Operation not permitted): An attempt was made to perform an operation that requires elevated privileges.
- **EPIPE** (Broken pipe): A write on a pipe, socket, or FIFO for which there is no process to read the data. Commonly encountered at the `net` and `http` layers, indicative that the remote side of the stream being written to has been closed.
- **ETIMEDOUT** (Operation timed out): A connect or send request failed because the connected party did not properly respond after a period of time. Usually encountered by `http` or `net` — often a sign that a `socket.end()` was not properly called.

Summary



Create responses that include:

- Headers
- Status code
- JSON data

The benefits of streams

- Memory efficiency
- Time efficiency

Error handling

- The "Error" class
- Common errors

