

- [\[Doc\]](#) `Process ()`
- [\[Doc\]](#) `Child Processes ()`
- [\[Doc\]](#) `Cluster ()`
- [\[Basic\]](#)
- [\[Basic\]](#)

Process, , ① , ② Node.js Process . , `ps -ef` .

UID	ID
PID	
PPID	
C	CPU
STIME	
TTY	
TIME	CPU
CMD	

APUE, Unix .

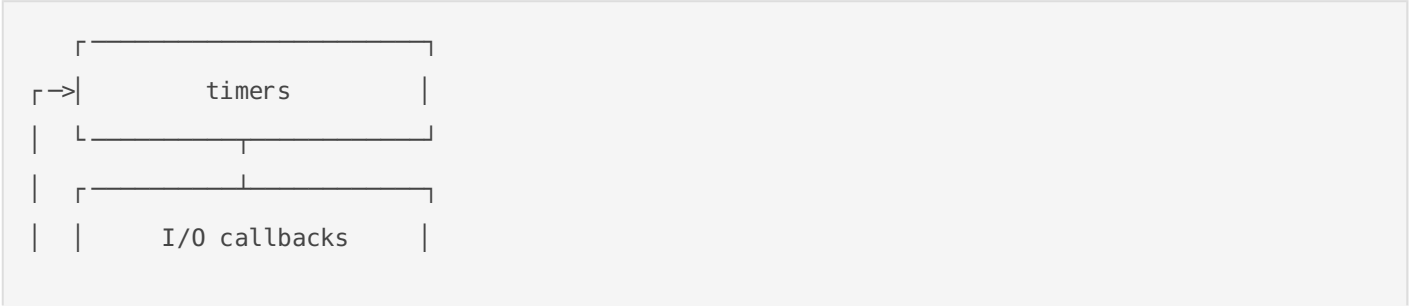
Process

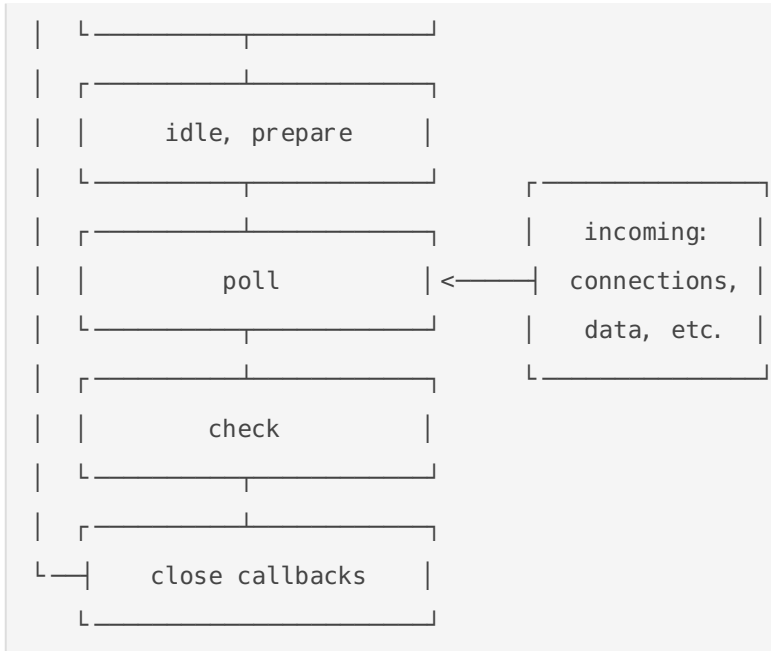
`Node``process` `console.log(process)` . ,`process` . :

- Usage
- /
- OS
-

process.nextTick

`process.nextTick` , , , .





`process.nextTick` Event loop , `nextTickQueue` "Tick", Queue .
(doge

```
function test() {  
  process.nextTick(() => test());  
}
```

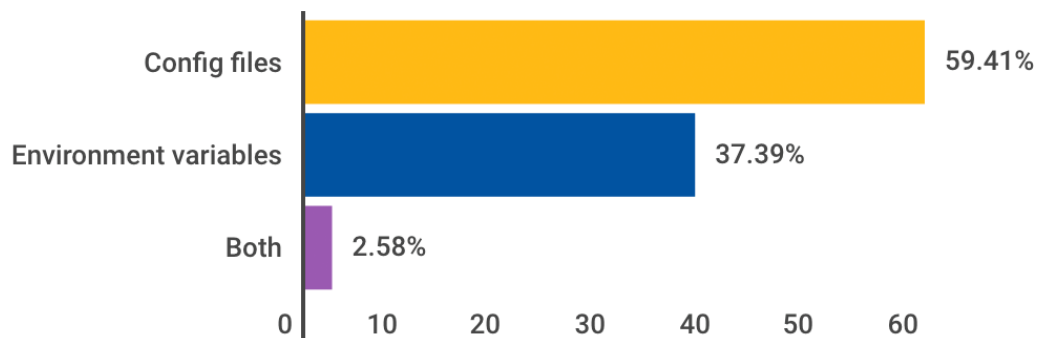
, ? ?

```
function test() {  
  setTimeout(() => test(), 0);  
}
```

. , , .

Environment variables or config files?

1126 respondents



Node.js Survey: survey.risingstack.com

`process.env`, `dotenv`, `node-config`, , , .

“ ? ?

, `process.cwd()` (current working directory), (),

`process.chdir()` . . , .

`process` `process.stderr`, `process.stdout` `process.stdin` , C/C++/Java `console.log` . ? , **console.log?**

C/C-`scanf`, C++ `cin`, Python `raw_input`).

, `top`, `ps`, `pstree` .

Child Process

(Child Process) `child_process` `e.js` `pomelo` , `cluster`) . .js .

“ child_process.fork POSIX fork ?

Node.js `child_process.fork()` Unix [fork\(\)](#) POSIX `fork` (waitpid), `child_pro`
Node.js , option .

- `spawn()`
 - `options.detached`
 - `options.stdio`
- `spawnSync()` `spawn`, ,
- `exec()` , ,
- `execSync()` `exec()`, , (stdout)
- `execFile()` ,
- `execFileSync()` `execFile()`, , exit code 0, throw Error
- `fork()` `spawn()`, `ChildProcess`

`exec/execSync` `bash` , , .

child.kill child.send

`child.kill` `child.send` . , IPC.

“ ? ?

`init` `wait()` `waitpid()` "PCB" " " , ,

Cluster

Cluster Node.js `child_process.fork()` , cluster IPC , [fork](#), . clus

```
const cluster = require('cluster'); // | |
const http = require('http'); // | |
const numCPUs = require('os').cpus().length; // | |

// | |
if (cluster.isMaster) { // | -|-----
  // Fork workers. // |
  for (var i = 0; i < numCPUs; i++) { // |
    cluster.fork(); // |
```

```

} // | (a.js)
cluster.on('exit', (worker) => { // |
  console.log(`${worker.process.pid} died`); // |
}); // |
} else { // |-----
  // Workers can share any TCP connection // |
  // In this case it is an HTTP server // |
  http.createServer((req, res) => { // |
    res.writeHead(200); // | (b.js)
    res.end('hello world\n'); // |
  }).listen(8000); // |
} // |-----

// | |
console.log('hello'); // | |
```

numCPUs, , , . , .

[a.js], [b.js], [node a.js] cluster.fork [node b.js]. cluster, cluster,

How It Works

worker child_process.fork() , IPC .

cluster .

(, windows), round-robin . , , socket

socket , socket worker, , worker .

, , 70% 8 2 , .

IPC (Inter-process communication) . :

PIPE	N	Y	Y	N
PIPE	N	Y	Y	N
	N	Y	Y	N
	N	Y	Y	Y
	N	Y	Y	Y
UNIX SOCKET	N	Y	Y	N

UNIX	SOCKET	Y	Y	N

Node.js IPC libuv , windows named pipe , *nix UDS (Unix Domain Socket)

socket , IPC socket , encode/decode

Node.js IPC ,

“ IPC , ? , IPC ?

child_process , `NODE_CHANNEL_FD` Node.js , fd IPC

(IPC) , IPC , IPC, IPC .

pm2 , , .

`&` , . session . tty , .

```
// (C )
void init_daemon()
{
    pid_t pid;
    int i = 0;

    if ((pid = fork()) == -1) {
        printf("Fork error !\n");
        exit(1);
    }

    if (pid != 0) {
        exit(0); //
    }

    setsid(); // ,
    if ((pid = fork()) == -1) {
        printf("Fork error !\n");
        exit(-1);
    }
    if (pid != 0) {
        exit(0); // ,
    }
}
```

```
        //      tty
    }
    chdir("/tmp");    //
    umask(0);        //
    for (; i < getdtablesize(); ++i) {
        close(i);    //
    }

    return;
}
```

Node.js

Revision #1

Created 19 July 2021 15:09:43 by

Updated 19 July 2021 15:12:37 by