YOU DON'T KNOW NODE.JS

QUICK GUIDE TO THE BEST FEATURES

 © CAPITAL ONE, 2015

SLIDES

Dear Kate,

Here's to the crazy ones. The misfits. The rebels. The troublemakers. The round pegs in the square holes. The ones who see things differently. They're not fond of rules. And they have no respect for the status quo. You can quote them, disagree with them, glorify or vilify them. About the only thing you can't do is ignore them. Because they change things. They push the human race forward. And while some may see them as the crazy ones, we see genius. Because the people who are crazy enough to think they can change the world, are the ones who do.

Take care.
John Appleseed

HTTPS://GITHUB.COM/AZAT-CO/YOU-DONT-KNOW-NODE

KEY TAKEAWAYS

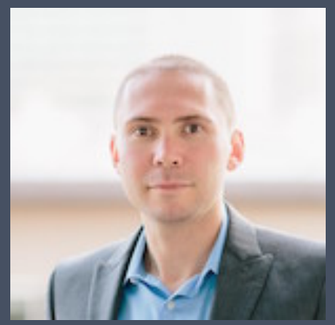
1. EVENT LOOP: BRUSH-UP ON THE CORE CONCEPT WHICH ENABLES THE NON-BLOCKING I/O
2. STREAMS AND BUFFERS: EFFECTIVE WAY TO WORK WITH DATA
3. PROCESS AND GLOBAL: HOW TO ACCESS MORE INFO

MORE KEY TAKEAWAYS

1. EVENT EMITTERS: CRASH COURSE IN THE EVENT-BASED PATTERN
2. CLUSTERS: FORK PROCESSES LIKE A PRO
3. ASYNCWRAP, DOMAIN AND UNCAUGHTEXCEPTION: HANDLING ASYNC ERRORS
4. C++ ADDONS: CONTRIBUTING TO THE CORE AND WRITING YOUR OWN C++ ADDONS

ABOUT PRESENTER

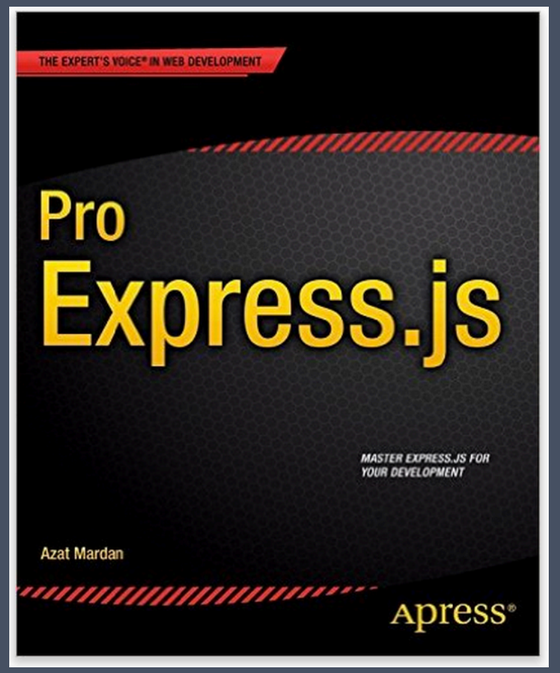
AZAT MARDAN

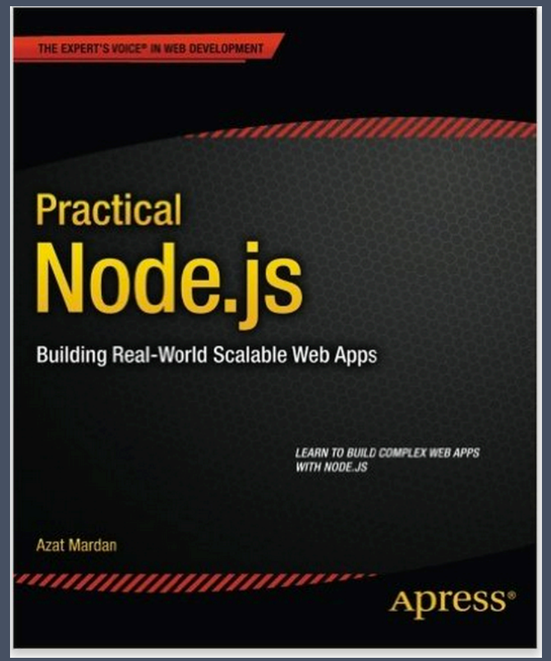


TWITTER: @AZAT_CO
EMAIL: HI@AZAT.CO
BLOG: WEBAPPLOG.COM

ABOUT PRESENTER

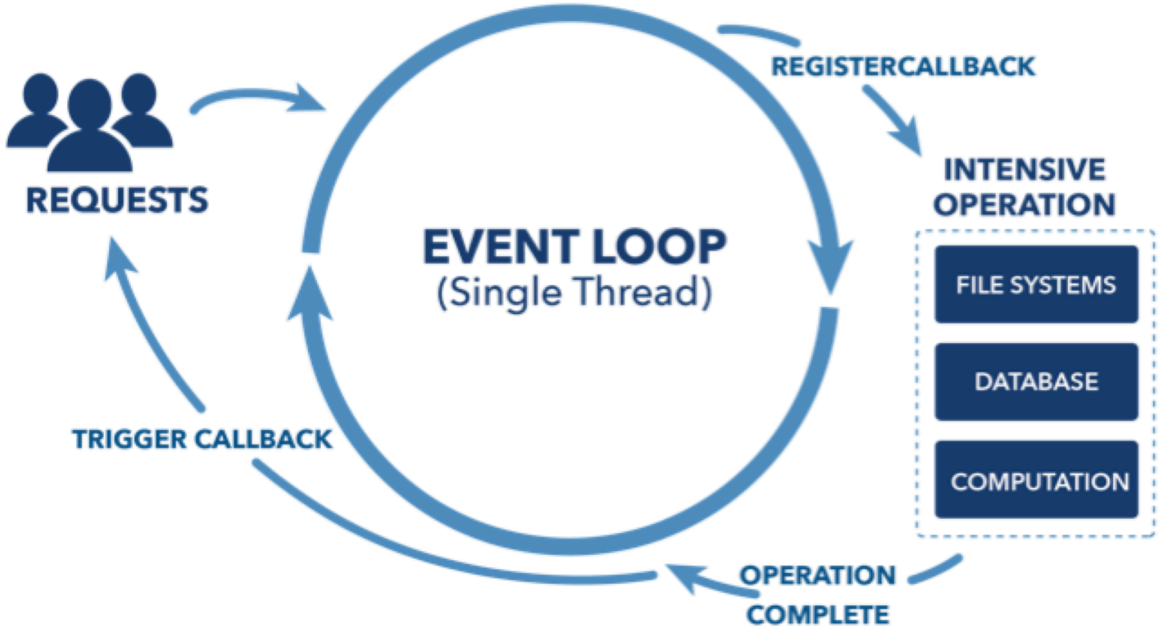
- TECHNOLOGY FELLOW AT CAPITAL ONE
- EXPERIENCE: FDIC. NIH. DOCUSIGN. HACKREACTOR AND STORIFY
- BOOKS: PRACTICAL NODE.JS. PRO EXPRESS.JS AND EXPRESS.JS API


THE EXPERT'S VOICE® IN WEB DEVELOPMENT
Pro
Express.js
MASTER EXPRESS.JS FOR YOUR DEVELOPMENT
Azat Mardan
Apress®


THE EXPERT'S VOICE® IN WEB DEVELOPMENT
Practical
Node.js
Building Real-World Scalable Web Apps
LEARN TO BUILD COMPLEX WEB APPS WITH NODE.JS
Azat Mardan
Apress®

EVENT LOOP

TECH | NON-BLOCKING I/O



BASIC EVENT LOOP EXAMPLE

```
System.out.println("Step: 1");
System.out.println("Step: 2");
Thread.sleep(1000);
System.out.println("Step: 3");
System.out.println("Step: 4");
```

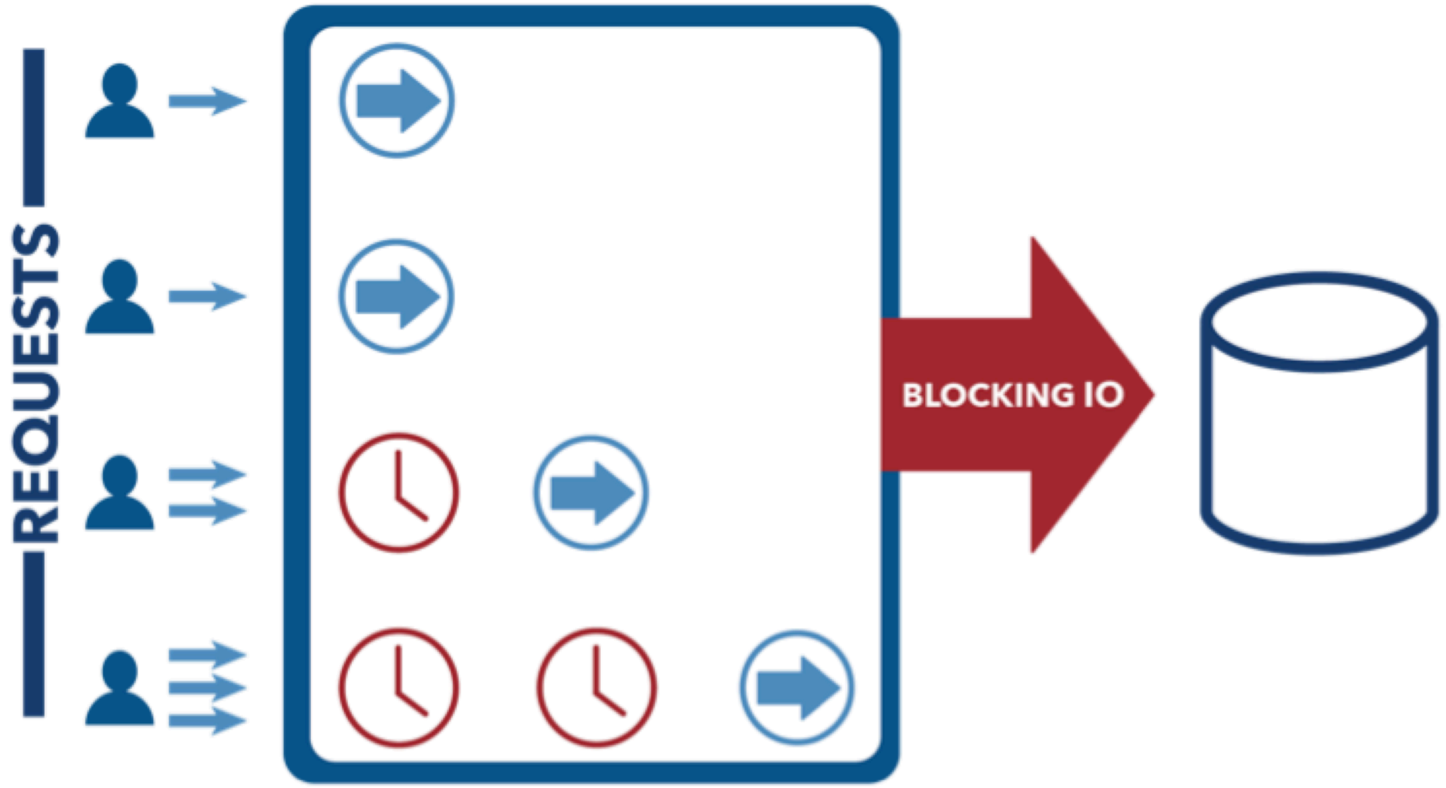
VS.

```
console.log('Step: 1')
setTimeout(function () {
  console.log('Step: 3')
}, 1000)
console.log('Step: 2')
```

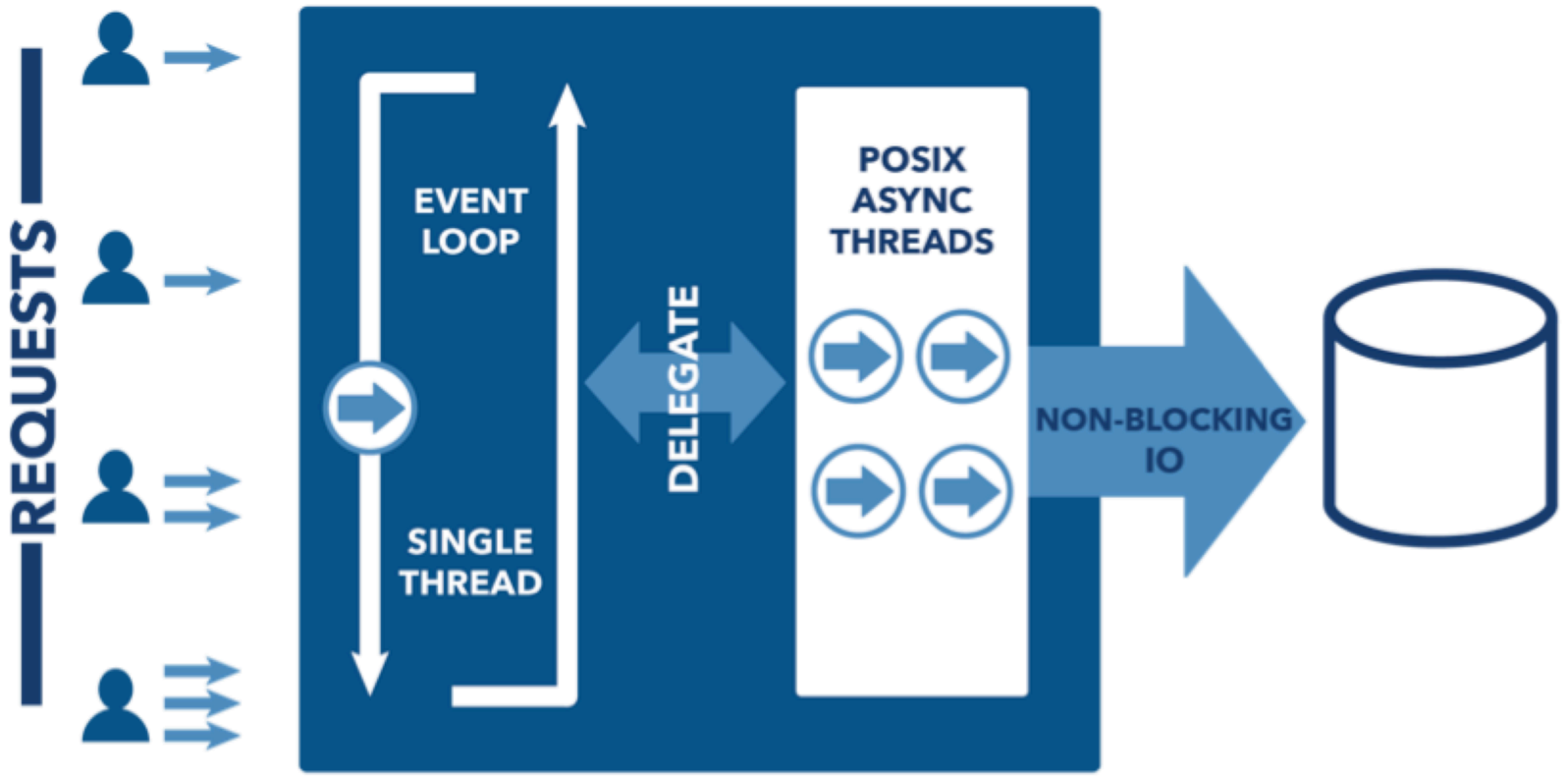
THINKING IN ASYNC CODE

```
console.log('Step: 1')
setTimeout(function () {
  console.log('Step: 3')
  // console.log('Step 5')
}, 1000);
console.log('Step: 2')
// console.log('Step 4')
```

TECH | # BLOCKING I/O



TECH | NON-BLOCKING I/O



IT'S STILL POSSIBLE TO WRITE BLOCKING CODE IN NODE.JS.

BLOCKING NODE.JS CODE

```
var fs = require('fs');

var contents = fs.readFileSync('accounts.txt','utf8');
console.log(contents);
console.log('Hello Capital One\n');

var contents = fs.readFileSync('ips.txt','utf8');
console.log(contents);
console.log('Hello SECON!');
```

NON-BLOCKING NODE.JS CODE

```
var fs = require('fs');

var contents = fs.readFile('accounts.txt','utf8', function(err,contents){
   console.log(contents);
});
console.log('Hello Capital One\n');


var contents = fs.readFile('ips.txt','utf8', function(err,contents){
   console.log(contents);
});
console.log("Hello SECON!");
```

STREAMS AND BUFFERS