Node.js

Server side of the future

What is node?

- נבנתה על בסיס מנוע V8 של chrome
- טובה מאוד לשרתים עם המון IO - כמו צ'אט, sockets, שרתי העלאת מדיה וכו'.
- מאוד מהירה, משום שהיא מבוססת על C
- None Blocking
- Single Threading
- Scalable

None Blocking

- למדתם במערכות הפעלה (ותלמדו במערכות זמן אמת) שיטות ואלגוריתמים לניהול מעבד.
- המטרה הייתה תמיד, להשאיר את המעבד כמה שיותר זמן פעיל, וכמה שפחות זמן "ישן" (idle)
- מה הייתה הבעיה העיקרית שאיתה צריך להתמודד?

None Blocking

- כדי להתמודד עם קריאות IO, Node משתמשת בשיטת תכנות מונחה אירועים.
- דוגמא:
- נרצה לבצע את הפעולות הבאות,
  - קריאה קובץ והדפסת תוכנו
  - קריאה לפונציה שמבצעת חישובים ללא קשר לקובץ

None Blocking

- Regular system code:
  - Read file
  - Print file
  - Do calculations
- Event Driven Code:
  - Read file
    - Once done reading, print it
  - Do Calculations

- מה שונה פה?

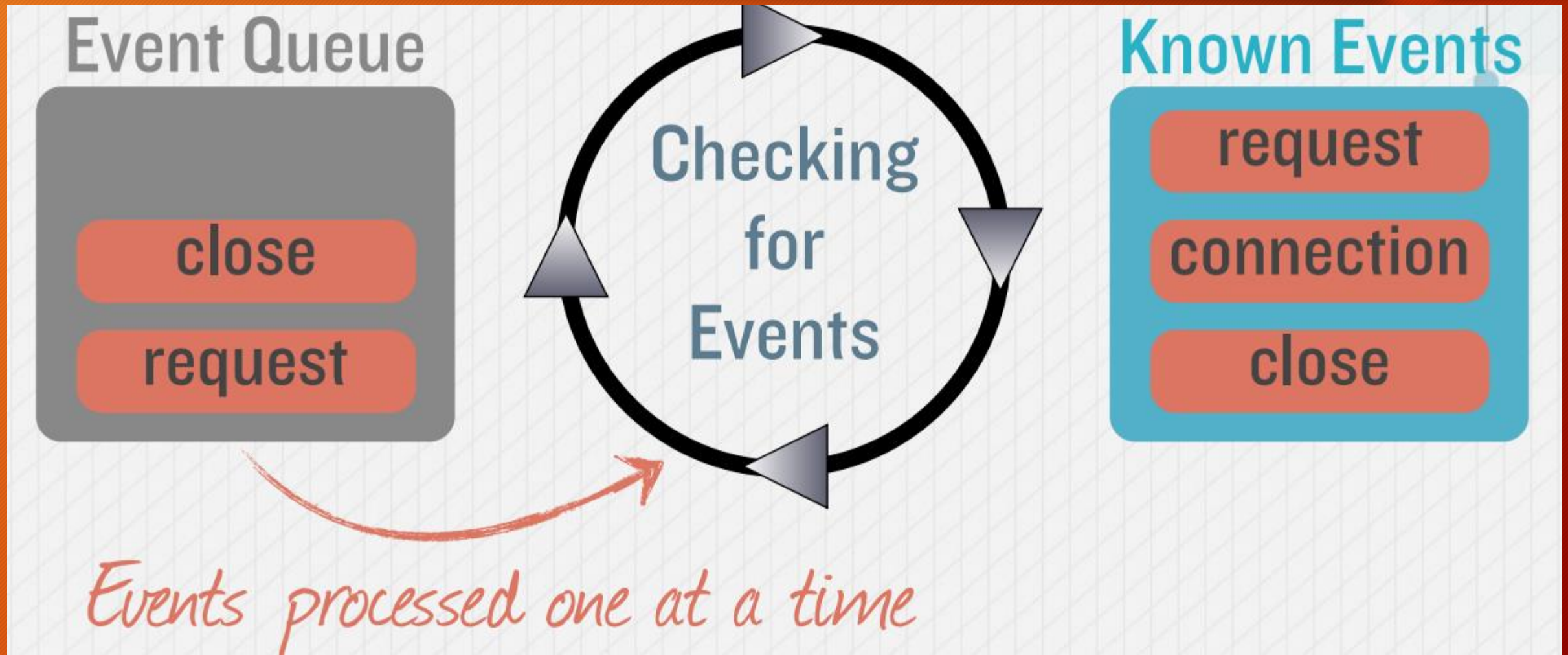
None Blocking

```
fs.readFile('/etc/hosts', function(err, contents) {
  console.log(contents);
});
console.log('Doing something else');
```

No More Idle

- בעצם בשיטה הזו, אין לנו מצב שבו המעבד נח
- הקריאה של הIO היא נפרדת, והקוד שלנו אינו מחכה שהקריאה תסתיים
- הקוד תמיד רץ!

Event Queue


Event Queue
close
request
Checking for Events
Known Events
request
connection
close
Events processed one at a time

Code Time

```
var http = require('http');  How we require modules

http.createServer(function(request, response) {
  response.writeHead(200);   Status code in header
  response.write("Hello, this is dog.");  Response body
  response.end(); Close the connection
}).listen(8080, function(){  Listen for connections on this port
  console.log('Listening on port 8080...');
});
```

Event shorter code

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


http.createServer(function(request, response) {
  response.writeHead(200);
  response.end("Hello, this is dog");
}).listen(8080);
```

```
$ node hello.js
```

Run the server

NPM

- מערכת התקנת תוספים, מהגדולות בעולם
- מגיעה עם התקנת Node
- בעזרתה נתקין הכל לשרת מהטרמינל (CMD)

NPM

- Installaion in CMD
  - npm install <packagename>
- Flags available
  - -g to install globally on the computer
  - --save to save to package.json in the server

Package.json

- קובץ אוטומטי שנוצר
- הקובץ מכיל את כל הספריות שהשרת דורש
- בכל התקנה של ספריה עם הדגל "save—" הספריה נרשמת בקובץ
- למה צריך אותו?
  - כאשר אנו מעלים את השרת לשירותים כמו גיטהאב, אנחנו לא רוצים להעלות את כל הספריות בנוסף, זה בזבוז של מקום ומיותר
  - לכן נעלה רק את השרת שלנו, יחד עם package.json
  - כאשר מישהו מוריד את הקוד, הוא עושה npm install בתקייה
  - פקודה זו תרוץ לבד על הקובץ הזה, ותתקין כל ספרייה שרשומה בו
  - השרת עובד!

Node_modules library

- תקייה אוטומטית שנוצרת אצלנו כשאר אנו מתקינים עם NPM
- התקייה מכילה את כל הספריות שאנו התקנו
- כאשר עושים require בקוד, השרת מחפש את הספריה שם.

Event Emitter

```
var events = require('events');
var EventEmitter = events.EventEmitter;

var chat = new EventEmitter();
var users = [], chatlog = [];

chat.on('message', function(message) {
  chatlog.push(message);
});

chat.on('join', function(nickname) {
  users.push(nickname);
});
```

Create Events

```
$("p").on("click", function(){ ... });
```

זוכרים את זה?

זה האזנה לאירוע click

בNode אנחנו יכולים ליצור אירועים משלנו

נאזין כך:

```
logger.on('error', function(message){
console.log('ERR: ' + message);
});
```

ונקרא לאירוע כך:

```
logger.emit('error', 'Spilled Milk');
```

Nodejs refs

- https://nodejs.org/api/
- http://www.tutorialspoint.com/nodejs/
- http://nodeschool.io/
- GOOGLE