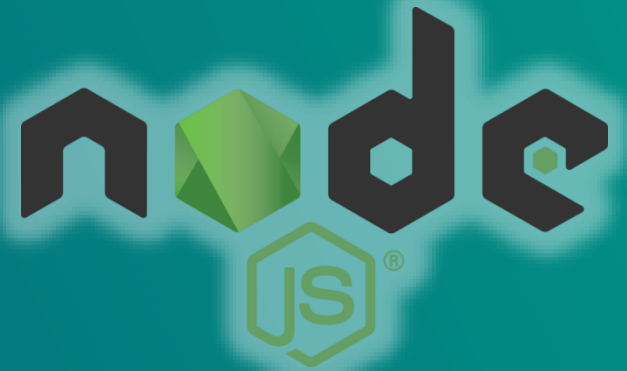


Back-End Development

Exploring

#2 Built-In Modules



Modules

Module is anything that can be loaded with *require()* in an NodeJS program, including:

- A folder with a *package.json* file containing a main field,
- A folder with an *index.js* file in it, or
- Just a JavaScript file.

NodeJS Built-In Modules

NodeJS has a set of built-in modules which you can use easily without any further installation.

- assert
- buffer
- cluster
- crypto
- dgram
- dns
- events
- fs
- http
- https
- net
- os
- path
- querystring
- readline
- stream
- string_decoder
- timers
- tls
- url
- util
- vm
- zlib



Timers Module

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

timers.setTimeout(function waktu() {
    console.log('Halo!');
}, 1000)

timers.setInterval(function waktu() {
    console.log('Ini loop!');
}, 2000)
```



Assert Module

unit testing

```
const assert = require('assert');  
var minum = {kopi:['luwak','hitam','susu']}
```

```
assert.equal(minum.kopi.length, 4);  
// ==
```

```
assert.strictEqual(minum.kopi.length, 4);  
// ===
```

```
assert.notEqual(minum.kopi.length, 3);  
// !=
```

```
assert.notStrictEqual(minum.kopi.length, 3);  
// !==
```



Assert Module

unit val testing

```
const assert = require('assert');  
var minum = {kopi:[12, 15, 20]};
```

```
assert.equal(minum.kopi[0], 'luwak');  
// ==
```

```
assert.strictEqual(minum.kopi[1] * 2, 30);  
// ===
```

```
assert.notEqual(minum.kopi[2], 20);  
// !=
```

```
assert.notStrictEqual(minum.kopi.length, 3);  
// !==
```



Url Module

```
var url = require('url');  
var link =  
  'http://lin.id/data.htm?tgl=2&bln=july';  
var x = url.parse(link, true);
```

```
console.log('Host = '+x.host);  
console.log('Path = '+x.pathname);  
console.log('Find = '+x.search);
```

```
var xdata = x.query;
```

```
console.log(xdata);  
console.log(xdata.tgl);  
console.log(xdata.bln);
```



```
const os = require('os');  
  
var namaCPU = os.hostname();  
var osTipe = os.type();  
var osPlatform = os.platform();  
var osRilis = os.release();  
var dirAwal = os.homedir();  
var ramSisa = os.freemem();  
var ramTotal = os.totalmem();
```

**Console.log semua var di atas!*


```
const os = require('os');  
var dataUser = os.userInfo();  
console.log(dataUser);
```

```
{ uid: -1,  
  gid: -1,  
  username: 'Windows 7',  
  homedir: 'C:\\Users\\Windows 7',  
  shell: null }
```

```
const os = require('os');  
var dataUser = os.userInfo();  
  
console.log(dataUser.uid);  
console.log(dataUser.gid);  
console.log(dataUser.username);  
console.log(dataUser.homedir);  
console.log(dataUser.shell);
```

```
const os = require('os');  
var dataCPU = os.cpus();  
console.log(dataCPU);
```

```
[ { model: 'AMD A6-4400M APU with Radeon(tm) HD Graphics',  
  speed: 2695,  
  times:  
    { user: 2016749,  
      nice: 0,  
      sys: 1205154,  
      idle: 15334835,  
      irq: 109824 } },  
  { model: 'AMD A6-4400M APU with Radeon(tm) HD Graphics',  
    speed: 2695,  
    times:  
      { user: 2102066,  
        nice: 0,  
        sys: 1426878,  
        idle: 15027014,  
        irq: 412856 } } ]
```

```
const os = require('os');  
var dataCPU = os.cpus();  
  
console.log(dataCPU[0].model);  
console.log(dataCPU[1].speed);  
console.log(dataCPU[1].times.user);
```



Events Module

```
// membuat event
```

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

```
eventKu.on('klik', function(){  
    console.log('Anda ngeklik!');  
});
```

```
eventKu.on('drag', function(){  
    console.log('Anda ngedrag!');  
});
```

Events Module

```
// mendata event
```

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

```
eventKu.on('klik', function(){  
    console.log('Anda ngeklik!'); }));
```

```
eventKu.on('drag', function(){  
    console.log('Anda ngedrag!'); }));
```

```
console.log(eventKu._eventsCount)  
console.log(eventKu.eventNames())
```

Events Module

```
// menjalankan event
```

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

```
eventKu.on('klik', function(){  
    console.log('Anda ngeklik!'); });
```

```
eventKu.on('drag', function(){  
    console.log('Anda ngedrag!'); });
```

```
eventKu.emit('klik');  
eventKu.emit('drag');
```

Events Module

```
// menjalankan event ber-parameter
```

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

```
var eventKu = new events.EventEmitter();
```

```
eventKu.on('klik', function(pesan){  
    console.log('Anda ngeklik! '+pesan);  
});
```

```
eventKu.on('drag', function(pesan){  
    console.log('Anda ngedrag! '+pesan); });
```

```
eventKu.emit('klik', ' OK!');
```

```
eventKu.emit('drag', ' Sip!');
```




FS Module

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

```
fs.writeFile('halo.txt', ' Halo!');
```

```
// membuat file halo.txt yg konten awalnya 'Halo!'
```

```
fs.appendFile('halo.txt', '\n Kuy!');
```

```
// menambah konten halo.txt dg '\n Kuy!'
```

***Coba appendFile dahulu, kemudian writeFile!**
Lihat outputnya!

FS Module

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

```
fs.writeFileSync('halo.txt', 'Halo!');  
// writeFileSync = writeFile synchronously
```

```
fs.appendFileSync('halo.txt', '\nYa!');  
// appendFileSync = appendFile synchronously
```

***Coba appendFileSync dahulu, kemudian writeFileSync!
Lihat outputnya!**

FS Module

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

var x =
fs.readFileSync('halo.txt');
console.log(x);

var y = fs.readFile('halo.txt',
    function(err, data){
        console.log(data)
    });
```

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

var x = fs.readFileSync('halo.txt',
    'utf8');
console.log(x);

var y = fs.readFile('halo.txt',
    'utf8',
    function(err, data){
        console.log(data)
    });
```

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

```
var x =  
fs.readFileSync('halo.txt');  
console.log(x.toString());
```

```
var y = fs.readFile('halo.txt',  
    function(err, data){  
        console.log(data.toString())  
    });
```

FS Module

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

```
fs.rename('halo.txt',  
'sapa.txt');
```

// ubah nama file 'halo.txt' menjadi 'sapa.txt'

```
fs.unlink('sapa.txt');
```

// menghapus file 'sapa.txt'

```
fs.mkdir('okay');
```

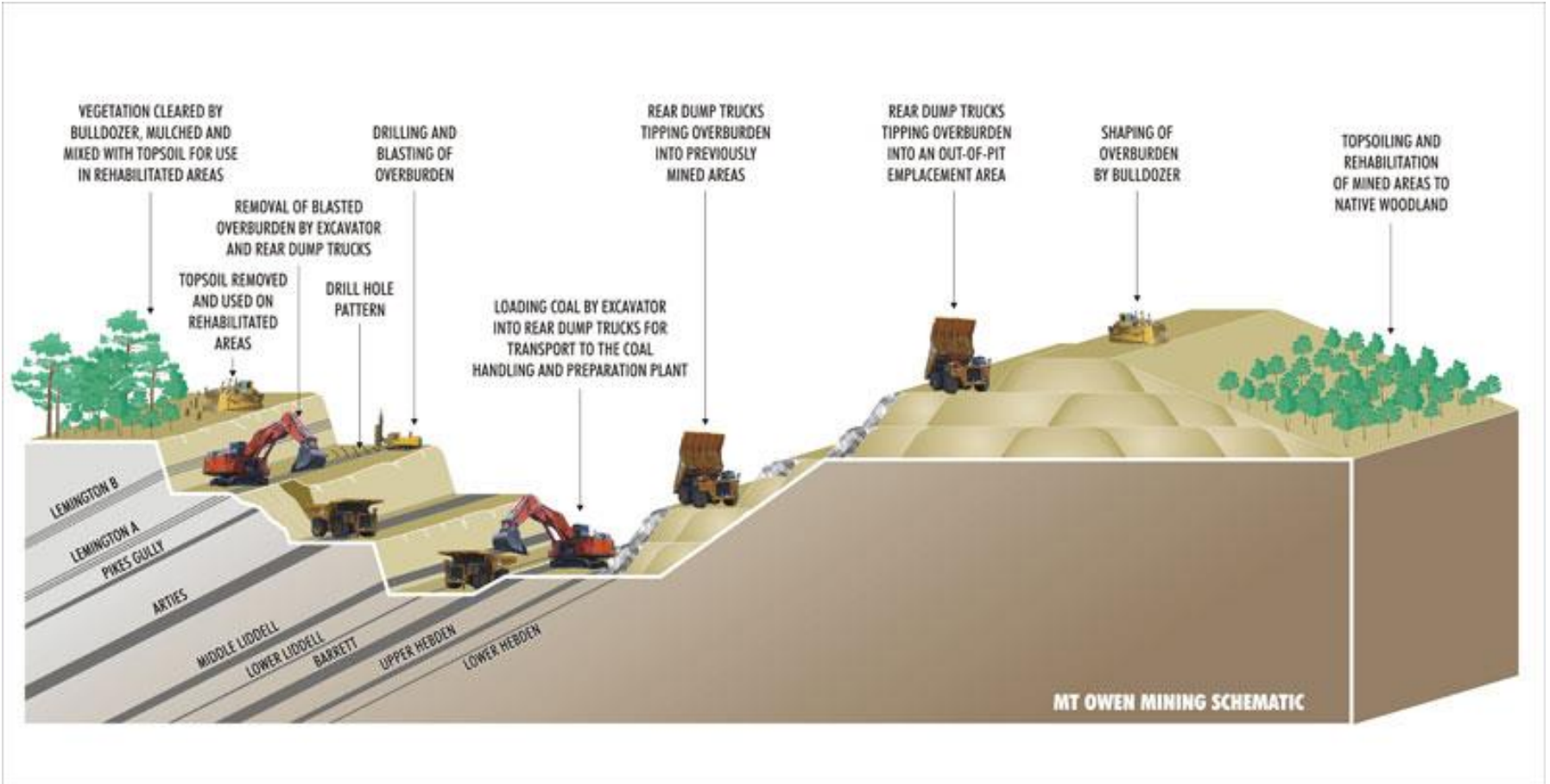
// membuat direktori/folder 'okay'

```
fs.rmdir('okay');
```

// menghapus direktori/folder 'okay'

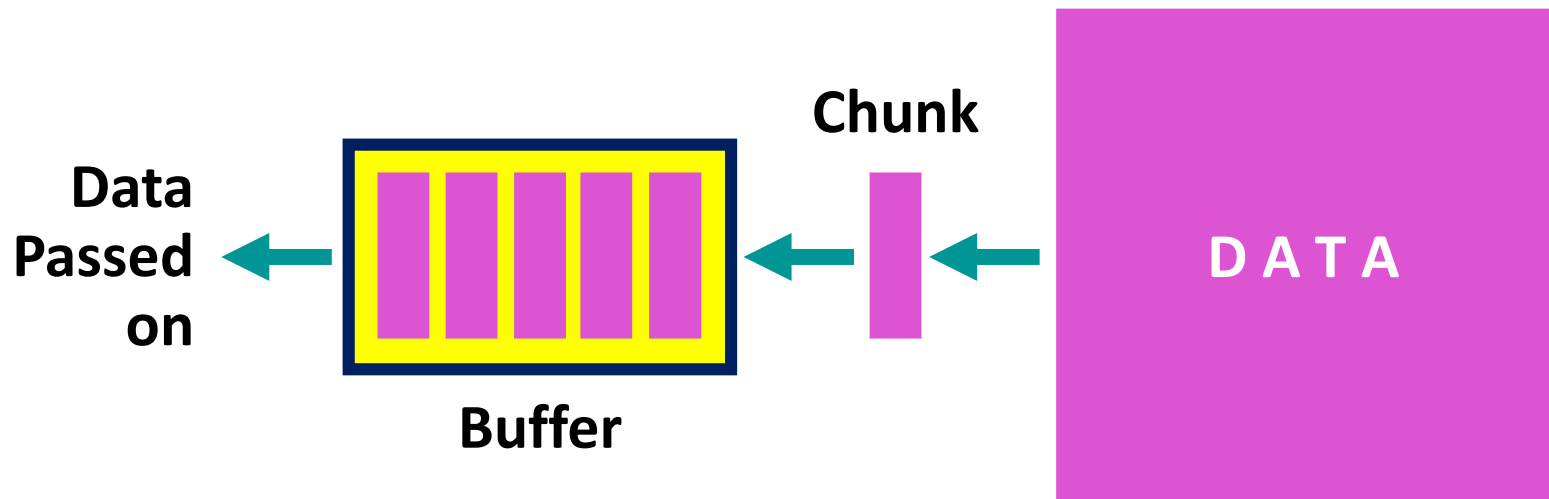
// folder yang akan dihapus harus kosong!

Buffer & Stream



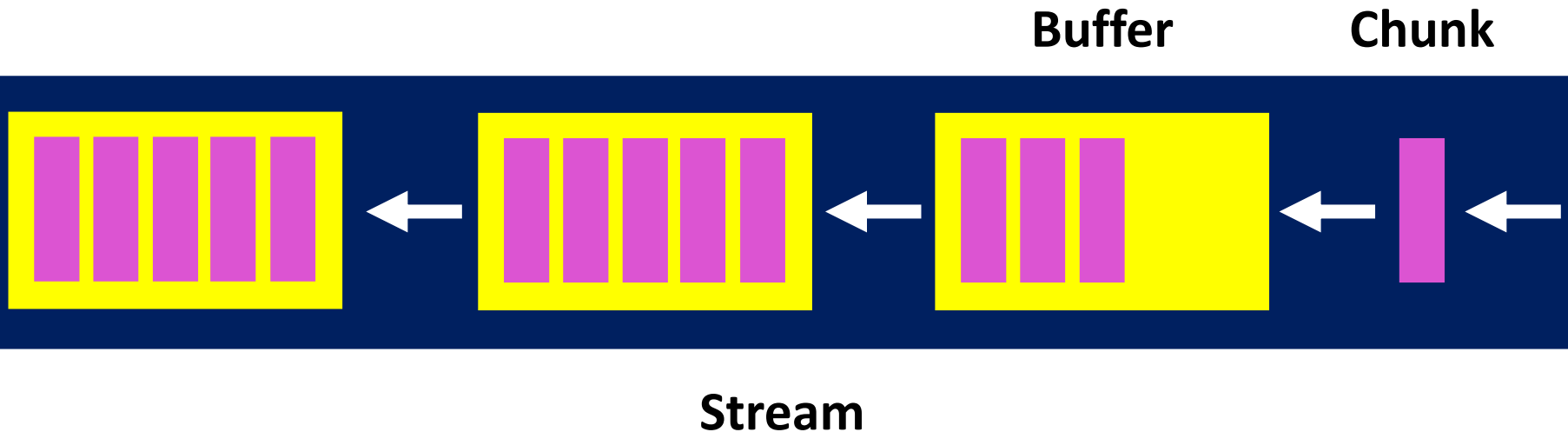
Buffer

- Buffer is temporary storage spot for a chunk of data that is being transferred from one place to another. Transfer small chunks of data at a time.



Stream

- Stream of data that flows over time from one place to another. Stream can increase our data transfer performance in Node.js.



***Buat halo.txt berisi 2000 baris @60 char!**

FS Read Stream

```
+ '/halo.txt',
```

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

```
var  BacaStream =  
 fs.createReadStream(__dirname+' /halo.txt',  
  'utf8');  
  
```

```

 bacaStream.on('data', function(potData){
    console.log('*** Potongan data masuk: ***');
    console.log(potData);
  })

```

• • • • •

[illegible][illegible]

*** Potongan data sudah masuk: ***

S

[illegible]

• • • • •

***Buat halo.txt berisi 2000 baris @60 char!**

FS Read Stream

```
+ '/halo.txt');  
^
```

```

    bacaStream.on('data', function(potData){
        console.log('*** Potongan data masuk: ***');
        console.log(potData.toString());
    })

```

[illegible]

FS Read Stream

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

var bacaStream =
fs.createReadStream(__dirname+'/halo.txt');

bacaStream.on('data', function(potData){
  console.log('*** Potongan data masuk: ***');
  console.log(potData);

  console.log(bacaStream.bytesRead);
  bacaStream.pause();
  setTimeout(function(){
    bacaStream.resume();
  }, 5000)
})
```

FS Read & Write Stream

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

var bacaStream =
fs.createReadStream(__dirname+'/halo.txt', 'utf8');

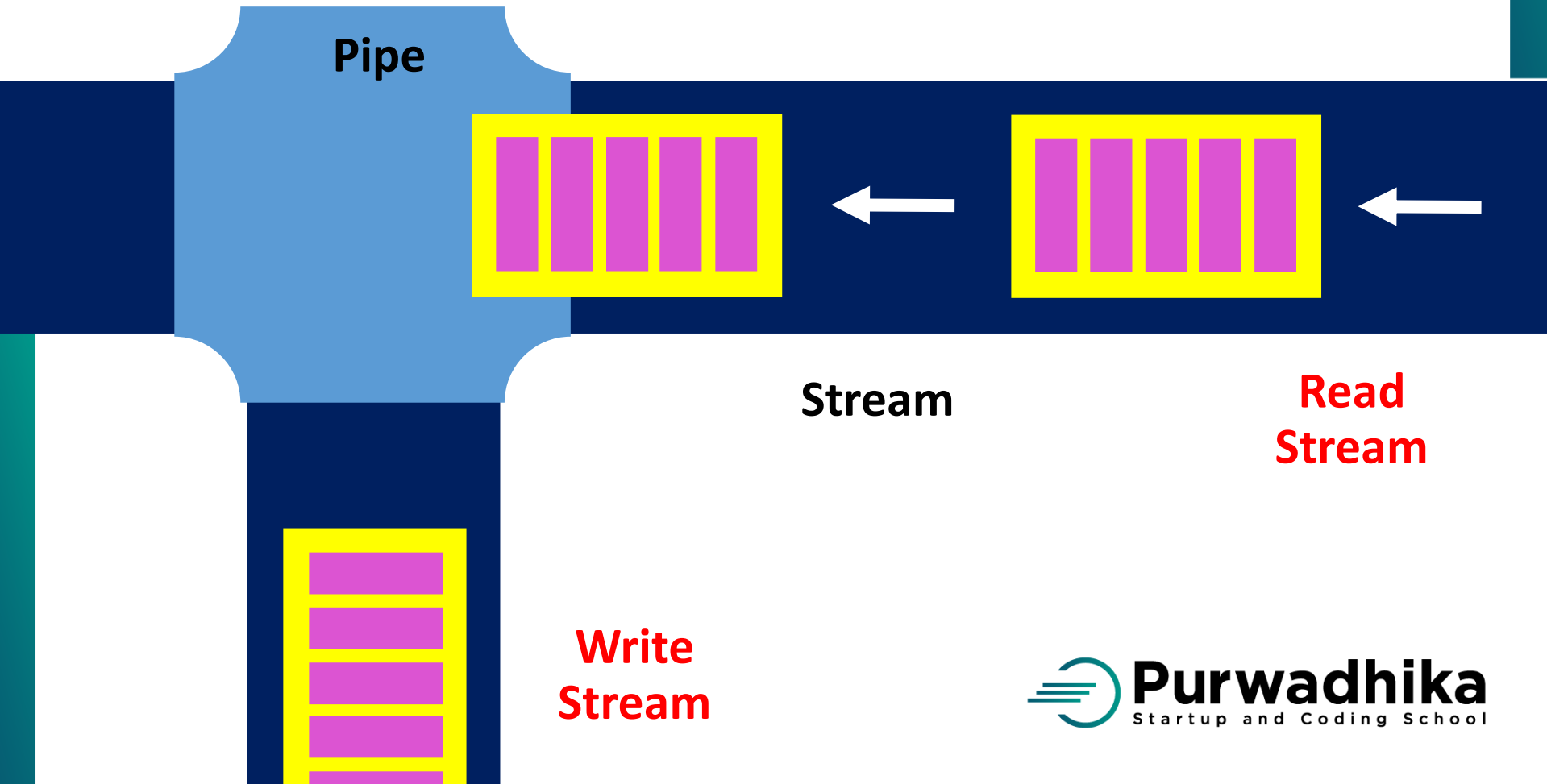
var tulisStream =
fs.createWriteStream(__dirname+'/halo2.txt');

bacaStream.on('data', function(potData){
    console.log('*** Potongan data masuk! ***');
    tulisStream.write(potData);
})
```

*Akan muncul file *halo2.txt* yg kontennya
Sama dengan hasil pembacaan *halo.txt*

Pipe

- Pipe takes data from a read stream and then “pipe” into a write stream.



FS Read & Write Stream

```
var fs = require('fs');  
  
var bacaStream =  
fs.createReadStream(__dirname+'/halo.txt', 'utf8');  
  
var tulisStream =  
fs.createWriteStream(__dirname+'/halo2.txt');  
  
bacaStream.pipe(tulisStream);
```

*Akan muncul file *halo2.txt* yg kontennya
Sama dengan hasil pembacaan *halo.txt*

Buatlah sebuah program yang ketika dijalankan akan menghasilkan file json berisi data json tentang spesifikasi OS/CPU user.

Gunakan module FS & OS!

#Solved!

```
const fs = require('fs');  
const os = require('os');  
  
var dataCPU = JSON.stringify(os.cpus());  
  
fs.writeFileSync('satu.json', dataCPU);  
  
console.log('Data sukses diterima!');
```

*Run kode di atas & akan muncul file *satu.json* berisi
Tentang spesifikasi OS/CPU user.

Back-End Development

Exploring

#2 Built-In Module

