Web Development with Node.js and MongoDB

Andrew Marcinkevičius

Video 2.4

Working with Streams

In this Video, we are going to take a look at…

- Streams
- Readable streams
- Writable streams
- Custom streams

Streams

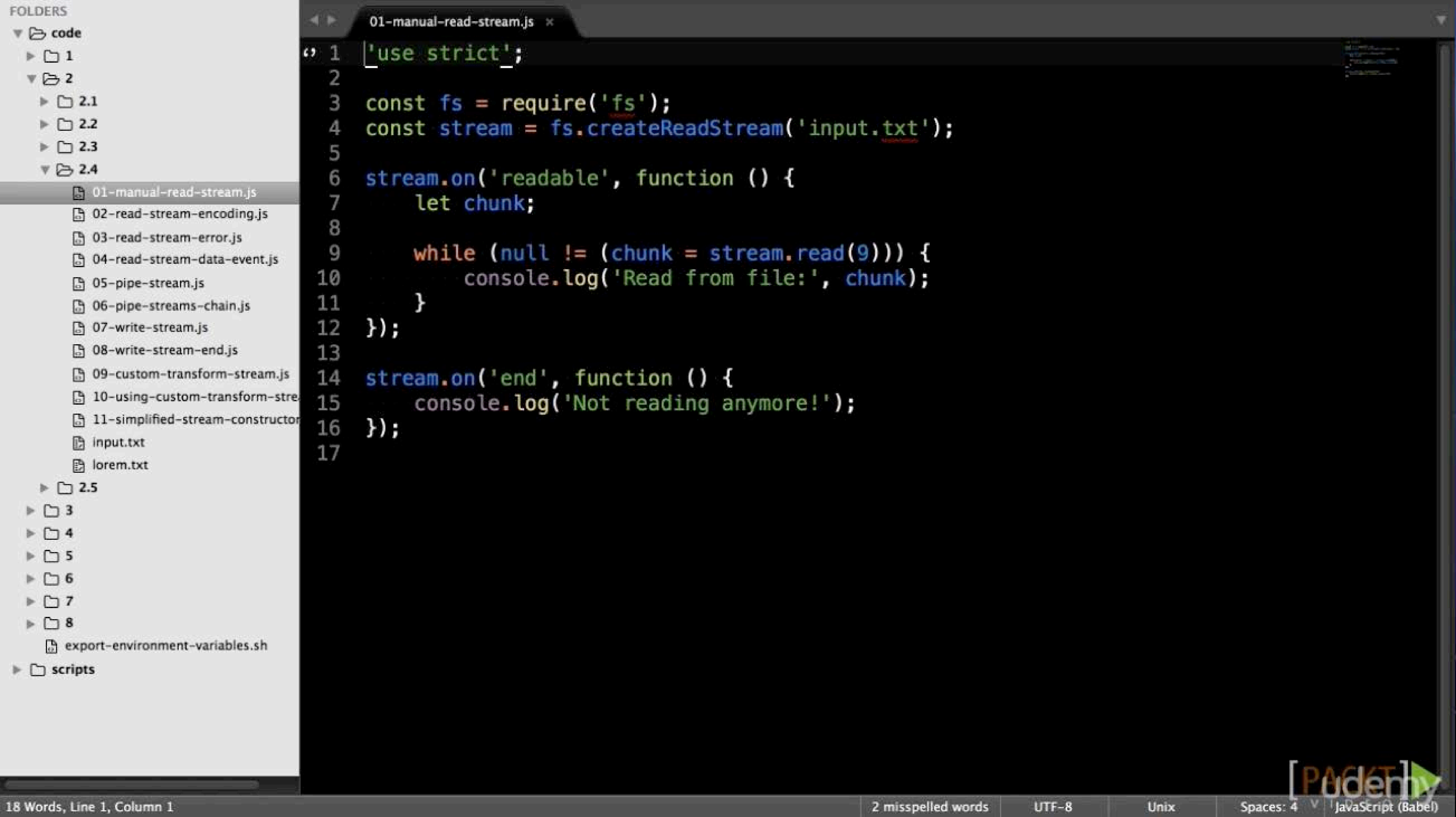
- Abstract interface
- Alternative way to access data
- Lower memory requirement
- Instances of EventEmitter
- Readable/Writable/Duplex

Readable Streams

- For receiving data
- Waits until you're ready to receive
- Flowing or paused mode
- Flowing mode: Receive as fast as possible
- Paused mode: Read manually

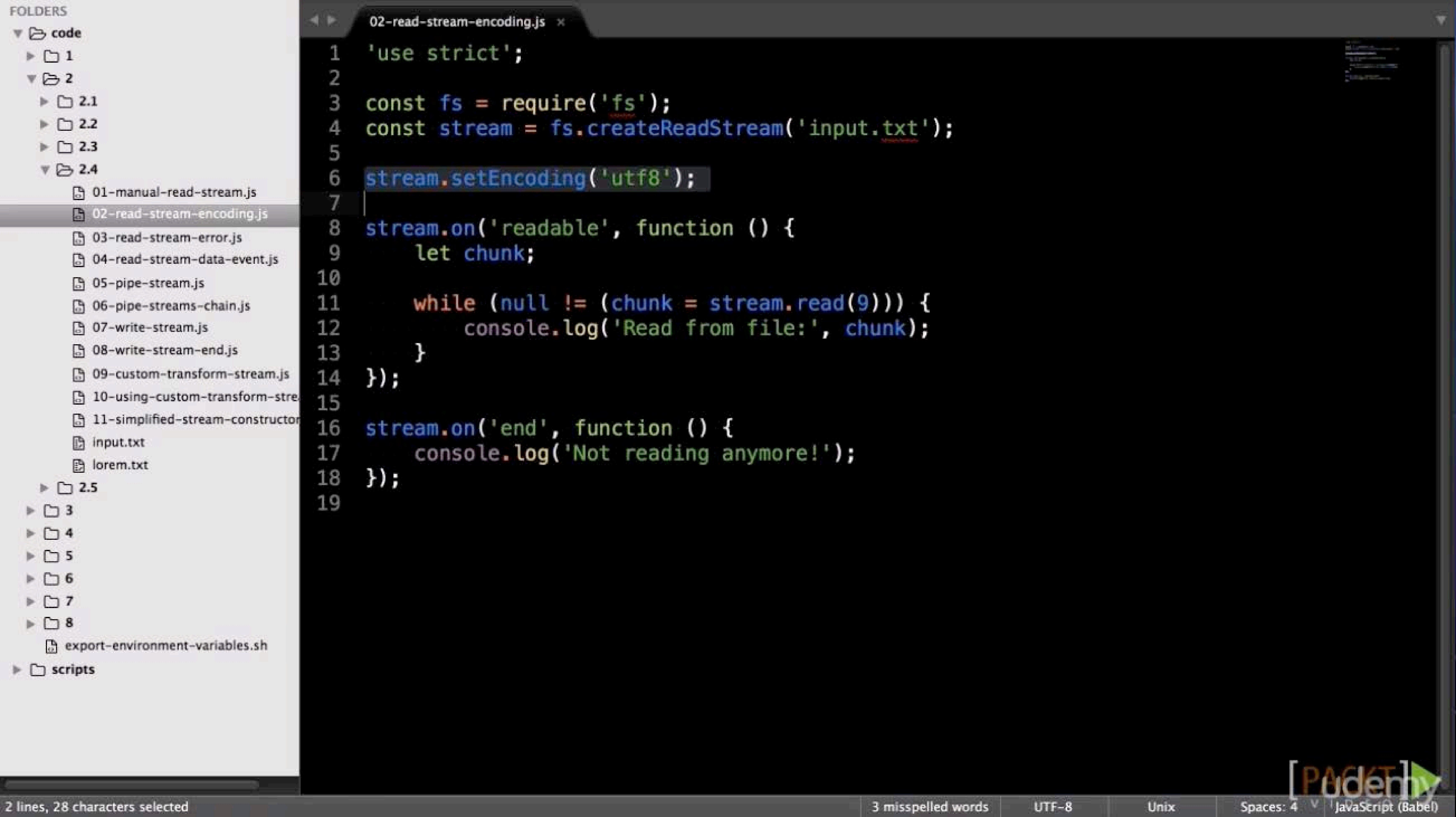
Readable Stream Modes

- Flowing mode
  - Add **data** event handler
  - Call **resume()** or **pipe()** method
- Paused mode
  - Call **pause()** method
  - Remove all **data** event handlers and pipe destinations



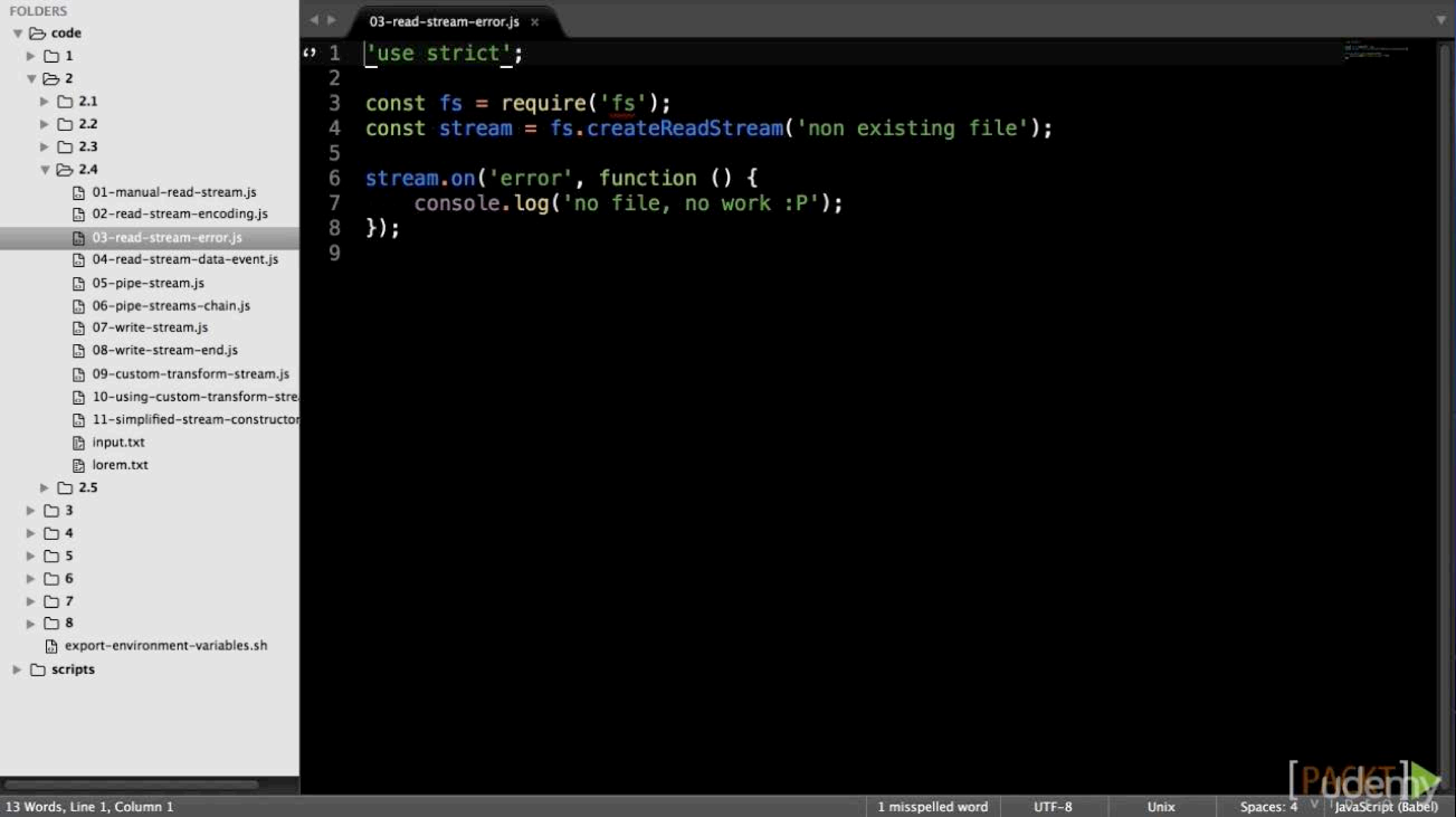
```
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('input.txt');

stream.on('readable', function () {
    let chunk;

    while (null != (chunk = stream.read(9))) {
        console.log('Read from file:', chunk);
    }
});

stream.on('end', function () {
    console.log('Not reading anymore!');
});
```

```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 01-manual-read-stream.js
Read from file: <Buffer 54 68 69 73 20 69 73 20 74>
Read from file: <Buffer 68 65 0a 69 6e 70 75 74 20>
Read from file: <Buffer 66 69 6c 65 20 66 6f 72 0a>
Read from file: <Buffer 73 74 72 65 61 6d 73 20 74>
Read from file: <Buffer 6f 70 69 63 0a>
Not reading anymore!

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$
```


FOLDERS
code
1
2
2.1
2.2
2.3
2.4
01-manual-read-stream.js
02-read-stream-encoding.js
03-read-stream-error.js
04-read-stream-data-event.js
05-pipe-stream.js
06-pipe-streams-chain.js
07-write-stream.js
08-write-stream-end.js
09-custom-transform-stream.js
10-using-custom-transform-stre
11-simplified-stream-constructor
input.txt
lorem.txt
2.5
3
4
5
6
7
8
export-environment-variables.sh
scripts
02-read-stream-encoding.js

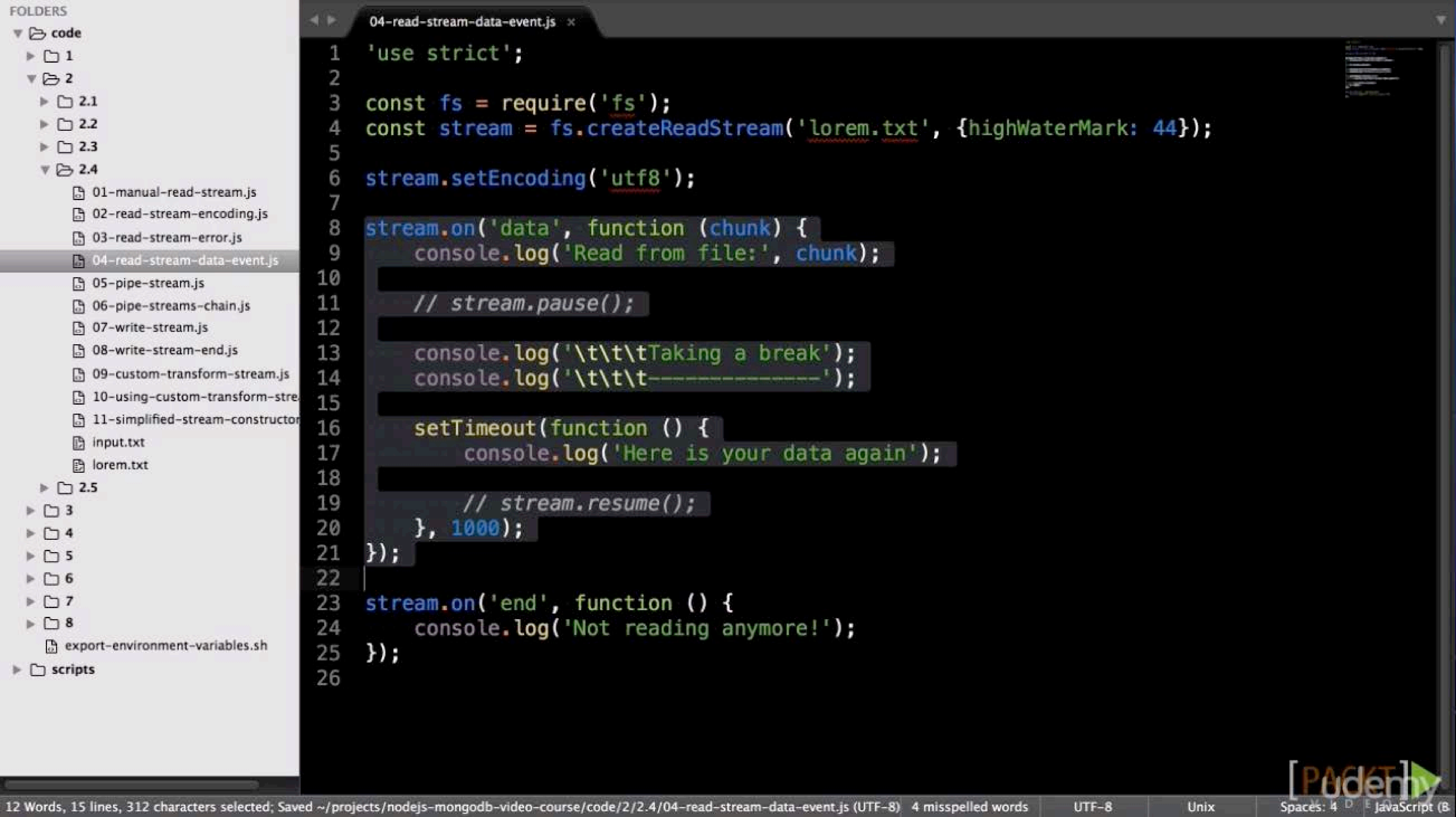
```javascript
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('input.txt');

stream.setEncoding('utf8');

stream.on('readable', function () {
    let chunk;

    while (null != (chunk = stream.read(9))) {
        console.log('Read from file:', chunk);
    }
});

stream.on('end', function () {
    console.log('Not reading anymore!');
});
```

```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 02-read-stream-encoding.js
Read from file: This is t
Read from file: he
input
Read from file: file for

Read from file: streams t
Read from file: opic

Not reading anymore!

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$
```



```
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('non existing file');

stream.on('error', function () {
    console.log('no file, no work :P');
});

```

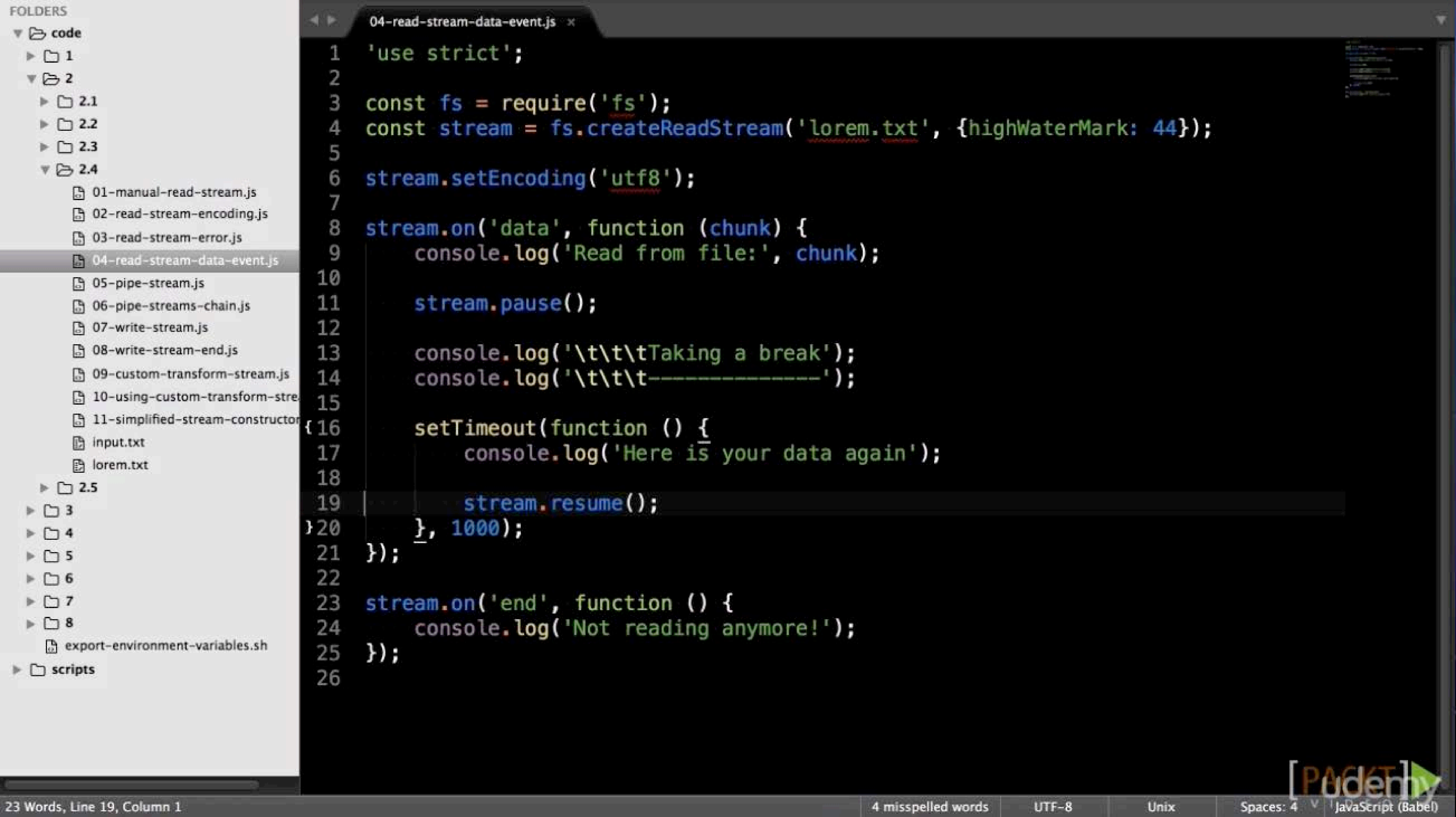
```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 03-read-stream-error.js
no file, no work :P

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ 
```



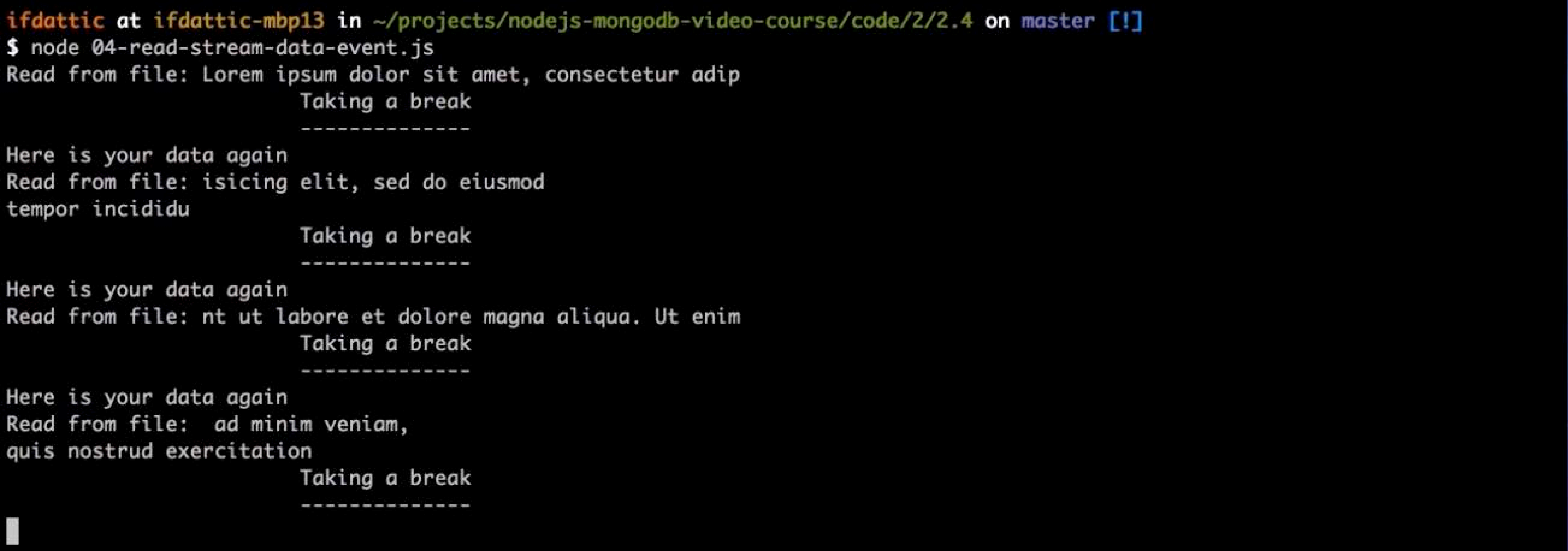
FOLDERS
code
1
2
2.1
2.2
2.3
2.4
01-manual-read-stream.js
02-read-stream-encoding.js
03-read-stream-error.js
04-read-stream-data-event.js
05-pipe-stream.js
06-pipe-streams-chain.js
07-write-stream.js
08-write-stream-end.js
09-custom-transform-stream.js
10-using-custom-transform-stre
11-simplified-stream-constructor
input.txt
lorem.txt
2.5
3
4
5
6
7
8
export-environment-variables.sh
scripts
04-read-stream-data-event.js

```javascript
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('lorem.txt', {highWaterMark: 44});

stream.setEncoding('utf8');

stream.on('data', function (chunk) {
    console.log('Read from file:', chunk);

    // stream.pause();

    console.log('\t\t\tTaking a break');
    console.log('\t\t\t-------------');

    setTimeout(function () {
        console.log('Here is your data again');

        // stream.resume();
    }, 1000);
});

stream.on('end', function () {
    console.log('Not reading anymore!');
});
```

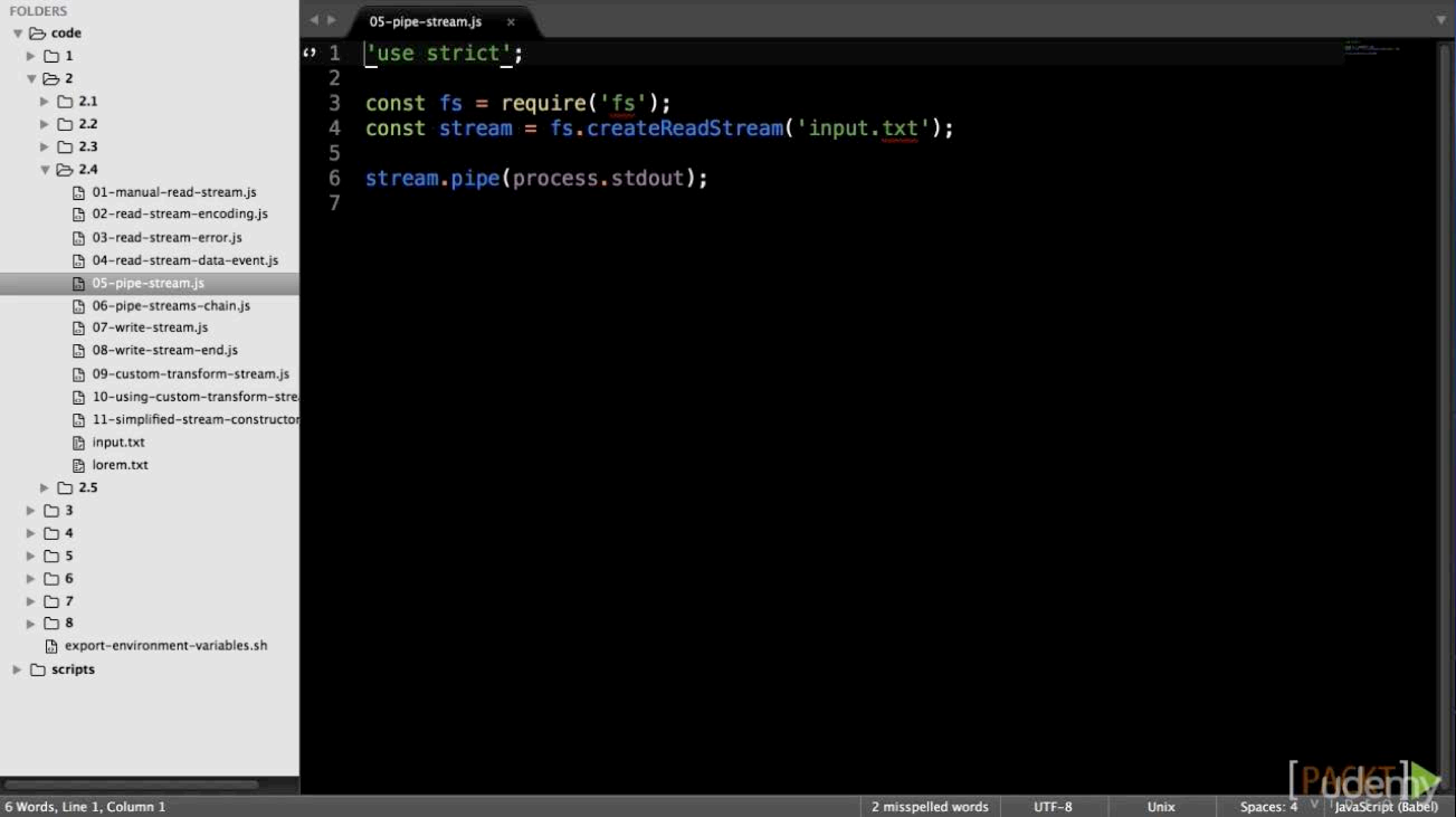
12 Words, 15 lines, 312 characters selected; Saved ~/projects/nodejs-mongodb-video-course/code/2/2.4/04-read-stream-data-event.js (UTF-8) 4 misspelled words UTF-8 Unix Spaces: 4 JavaScript (B


FOLDERS
code
1
2
2.1
2.2
2.3
2.4
01-manual-read-stream.js
02-read-stream-encoding.js
03-read-stream-error.js
04-read-stream-data-event.js
05-pipe-stream.js
06-pipe-streams-chain.js
07-write-stream.js
08-write-stream-end.js
09-custom-transform-stream.js
10-using-custom-transform-stre
11-simplified-stream-constructor
input.txt
lorem.txt
2.5
3
4
5
6
7
8
export-environment-variables.sh
scripts
04-read-stream-data-event.js

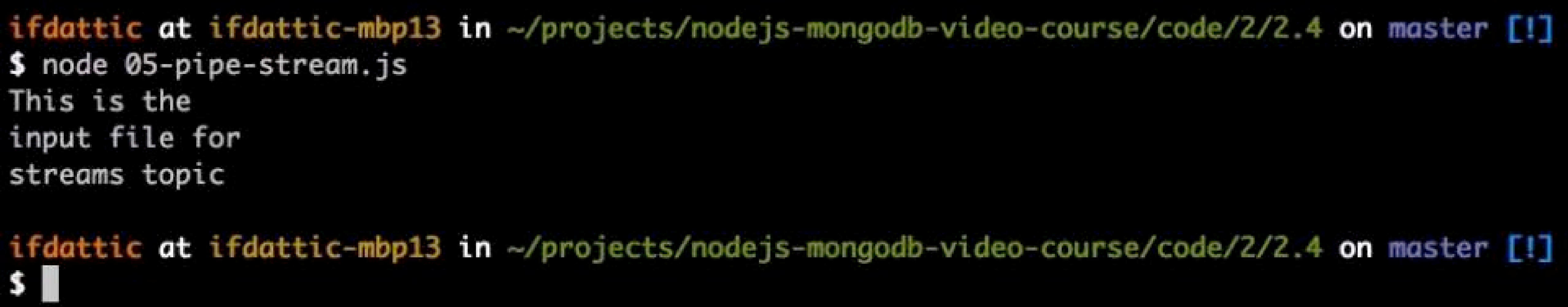
```
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('lorem.txt', {highWaterMark: 44});

stream.setEncoding('utf8');

stream.on('data', function (chunk) {
    console.log('Read from file:', chunk);

    stream.pause();

    console.log('\t\t\tTaking a break');
    console.log('\t\t\t--------------');

    setTimeout(function () {
        console.log('Here is your data again');

        stream.resume();
    }, 1000);
});

stream.on('end', function () {
    console.log('Not reading anymore!');
});
```



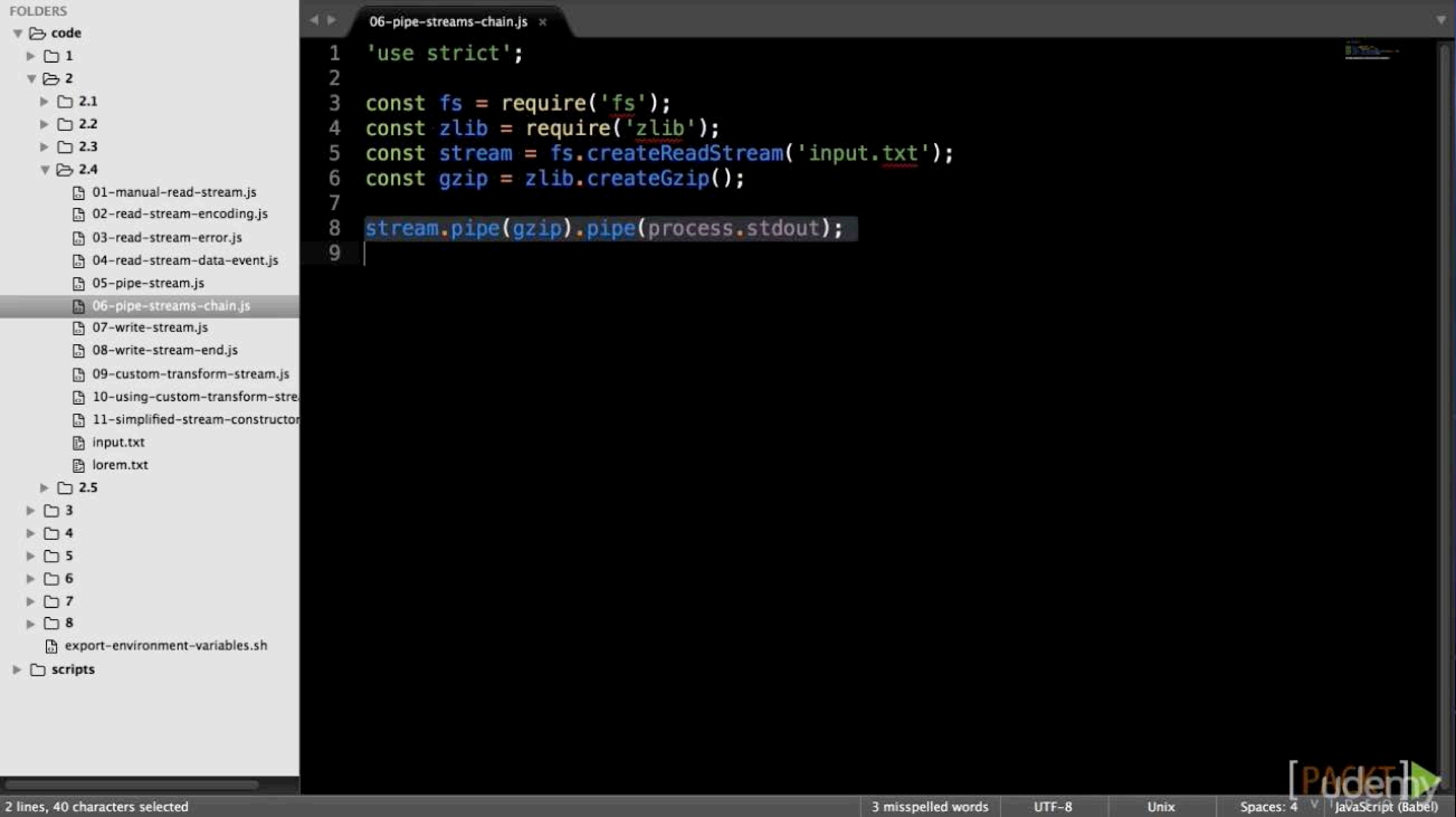
```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 04-read-stream-data-event.js
Read from file: Lorem ipsum dolor sit amet, consectetur adip
                        Taking a break
                        --------------
Here is your data again
Read from file: isicing elit, sed do eiusmod
tempor incididu
                        Taking a break
                        --------------
Here is your data again
Read from file: nt ut labore et dolore magna aliqua. Ut enim
                        Taking a break
                        --------------
Here is your data again
Read from file:  ad minim veniam,
quis nostrud exercitation
                        Taking a break
                        --------------
```



```
'use strict';

const fs = require('fs');
const stream = fs.createReadStream('input.txt');

stream.pipe(process.stdout);

```

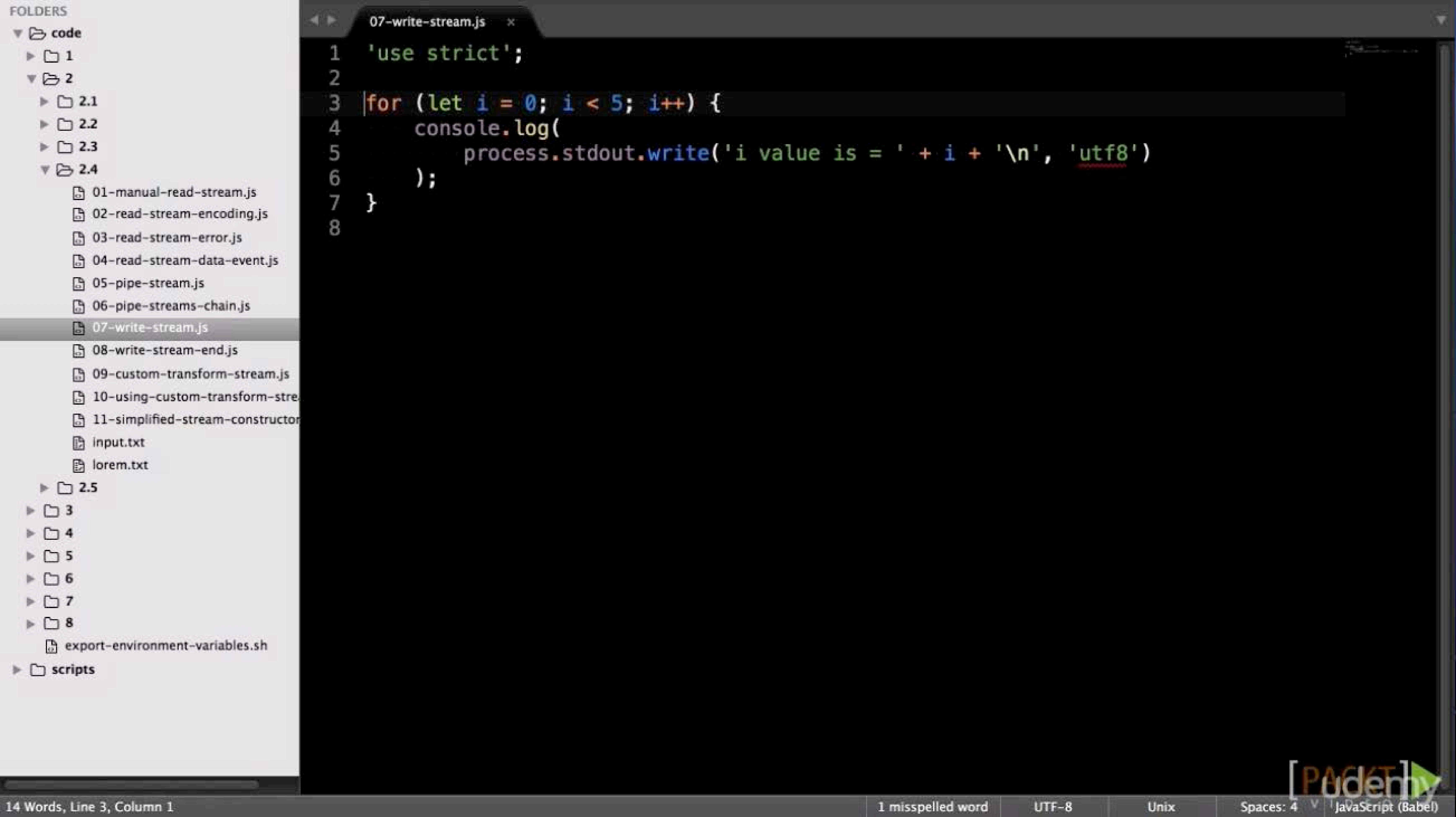


```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 05-pipe-stream.js
This is the
input file for
streams topic

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ 
```

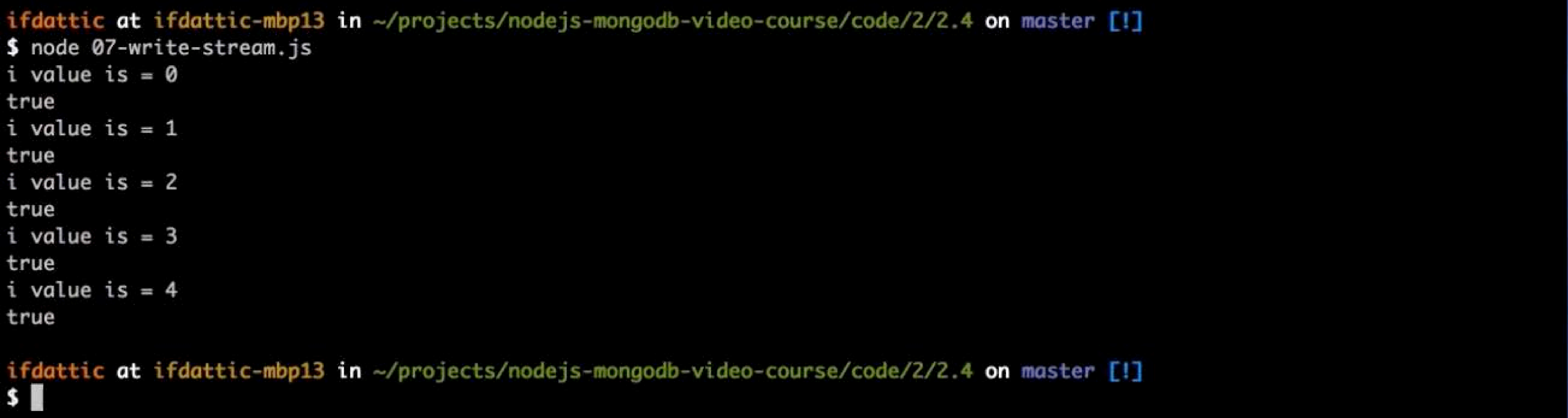


```
'use strict';

const fs = require('fs');
const zlib = require('zlib');
const stream = fs.createReadStream('input.txt');
const gzip = zlib.createGzip();

stream.pipe(gzip).pipe(process.stdout);

```

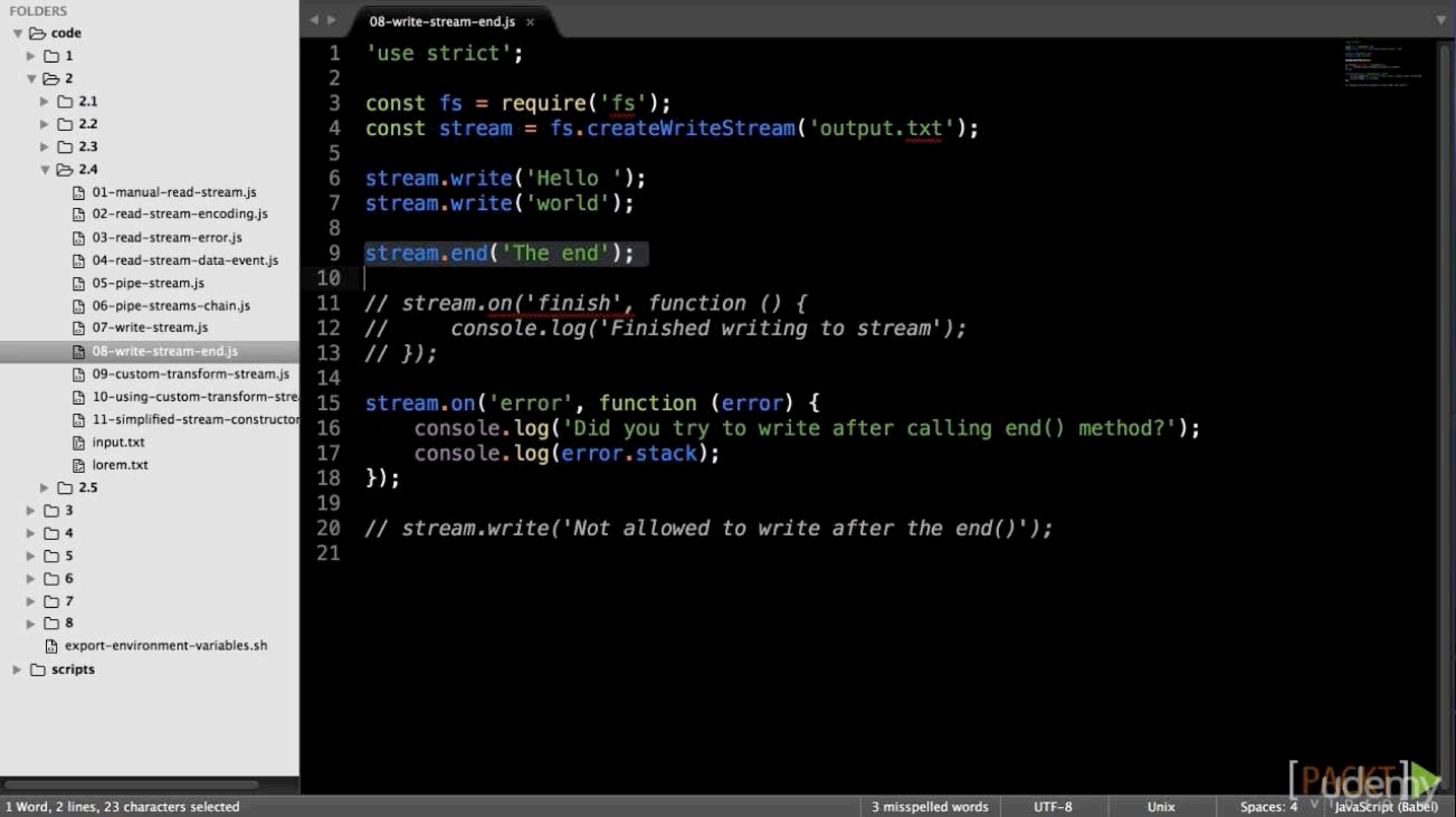
```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 06-pipe-streams-chain.js
�
��,V���T�����T���"�g��\�T~Af2��B)
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ 
```



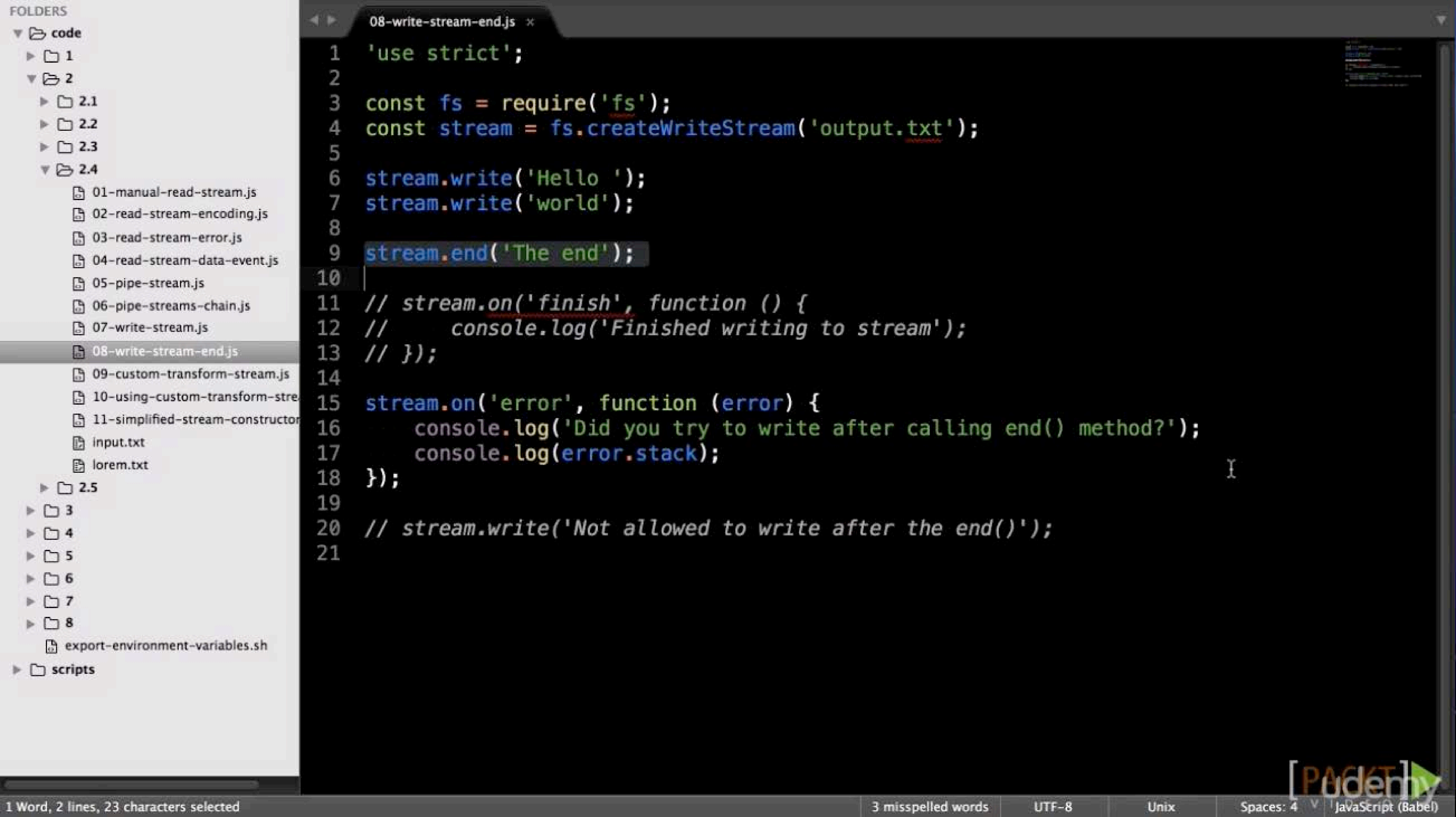
```javascript
'use strict';

for (let i = 0; i < 5; i++) {
    console.log(
        process.stdout.write('i value is = ' + i + '\n', 'utf8')
    );
}

```



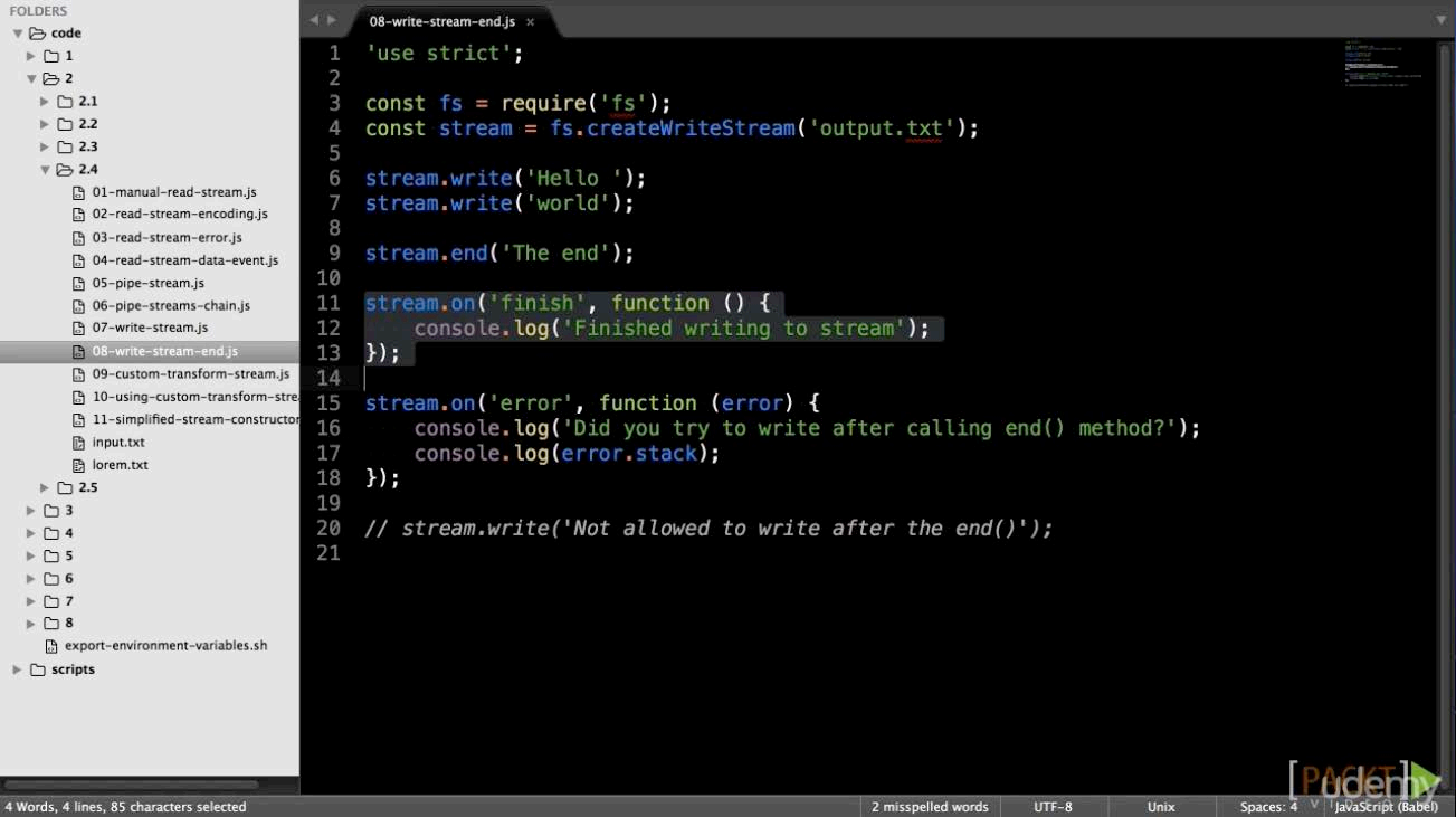
```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 07-write-stream.js
i value is = 0
true
i value is = 1
true
i value is = 2
true
i value is = 3
true
i value is = 4
true

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ 
```



```javascript
'use strict';

const fs = require('fs');
const stream = fs.createWriteStream('output.txt');

stream.write('Hello ');
stream.write('world');

stream.end('The end');

// stream.on('finish', function () {
//     console.log('Finished writing to stream');
// });

stream.on('error', function (error) {
    console.log('Did you try to write after calling end() method?');
    console.log(error.stack);
});

// stream.write('Not allowed to write after the end()');
```

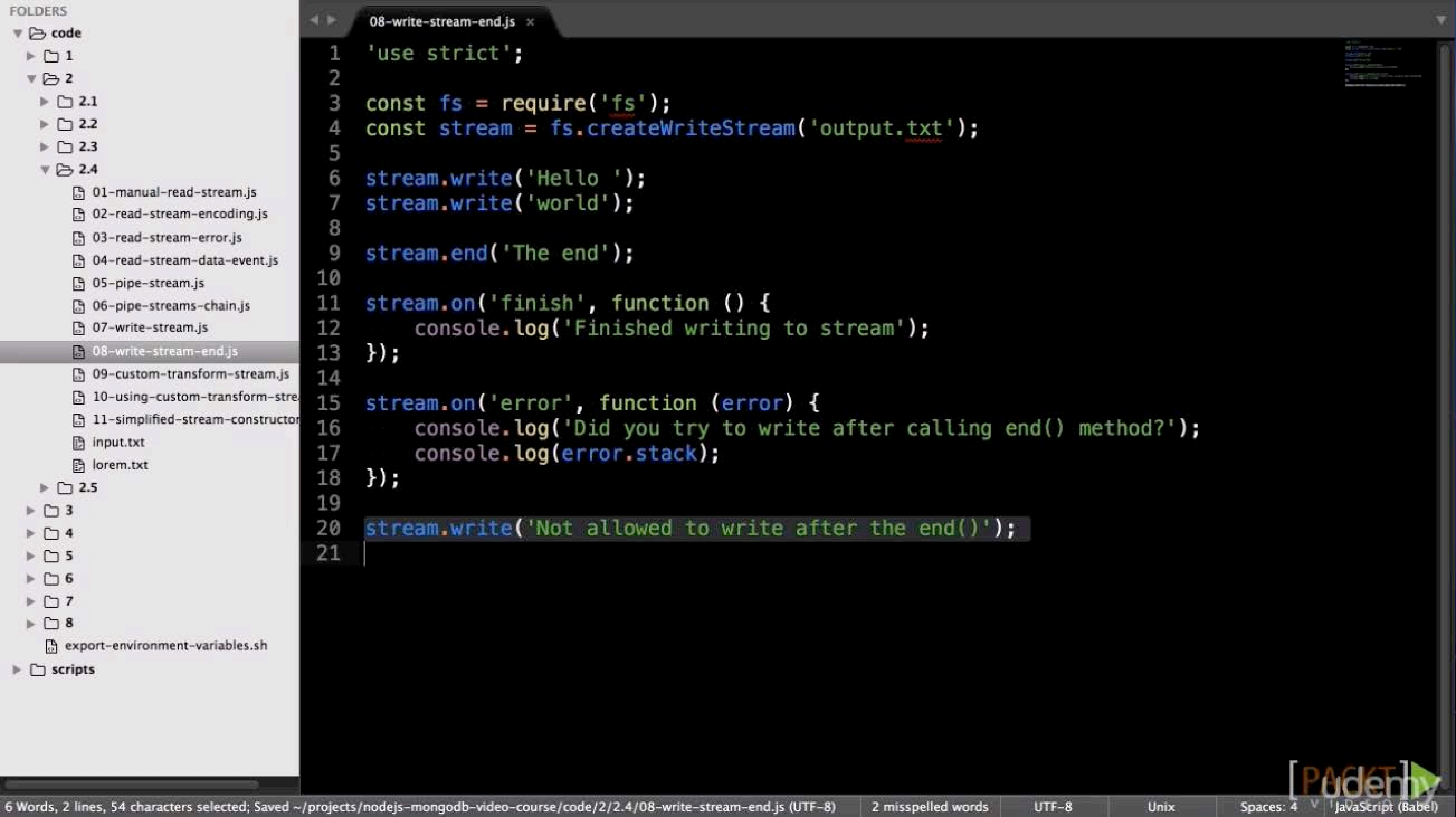


```javascript
'use strict';

const fs = require('fs');
const stream = fs.createWriteStream('output.txt');

stream.write('Hello ');
stream.write('world');

stream.end('The end');

// stream.on('finish', function () {
//     console.log('Finished writing to stream');
// });

stream.on('error', function (error) {
    console.log('Did you try to write after calling end() method?');
    console.log(error.stack);
});

// stream.write('Not allowed to write after the end()');
```



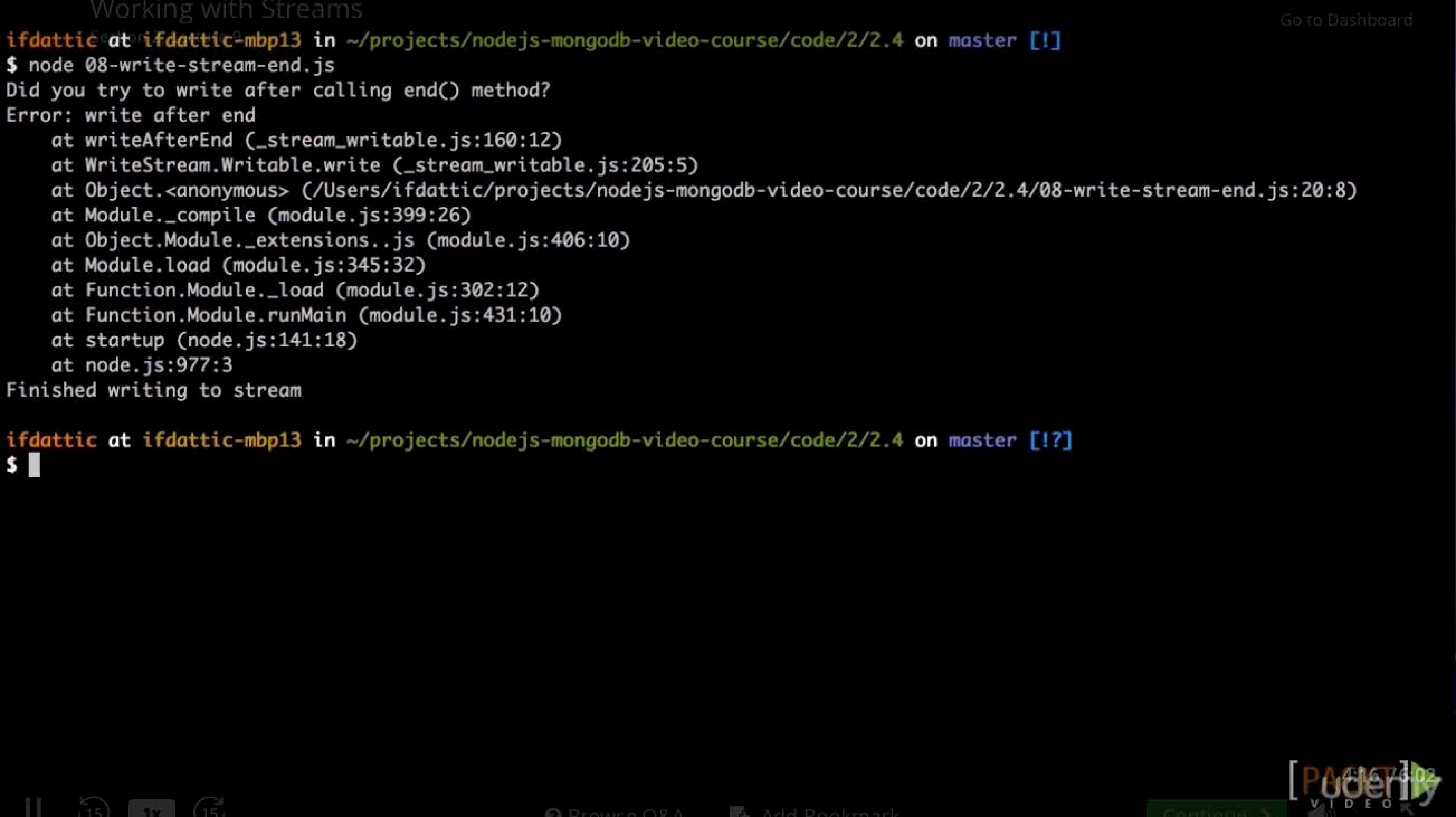
```javascript
'use strict';

const fs = require('fs');
const stream = fs.createWriteStream('output.txt');

stream.write('Hello ');
stream.write('world');

stream.end('The end');

stream.on('finish', function () {
    console.log('Finished writing to stream');
});

stream.on('error', function (error) {
    console.log('Did you try to write after calling end() method?');
    console.log(error.stack);
});

// stream.write('Not allowed to write after the end()');

```

```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ nod
```



```
'use strict';

const fs = require('fs');
const stream = fs.createWriteStream('output.txt');

stream.write('Hello ');
stream.write('world');

stream.end('The end');

stream.on('finish', function () {
    console.log('Finished writing to stream');
});

stream.on('error', function (error) {
    console.log('Did you try to write after calling end() method?');
    console.log(error.stack);
});

stream.write('Not allowed to write after the end()');

```

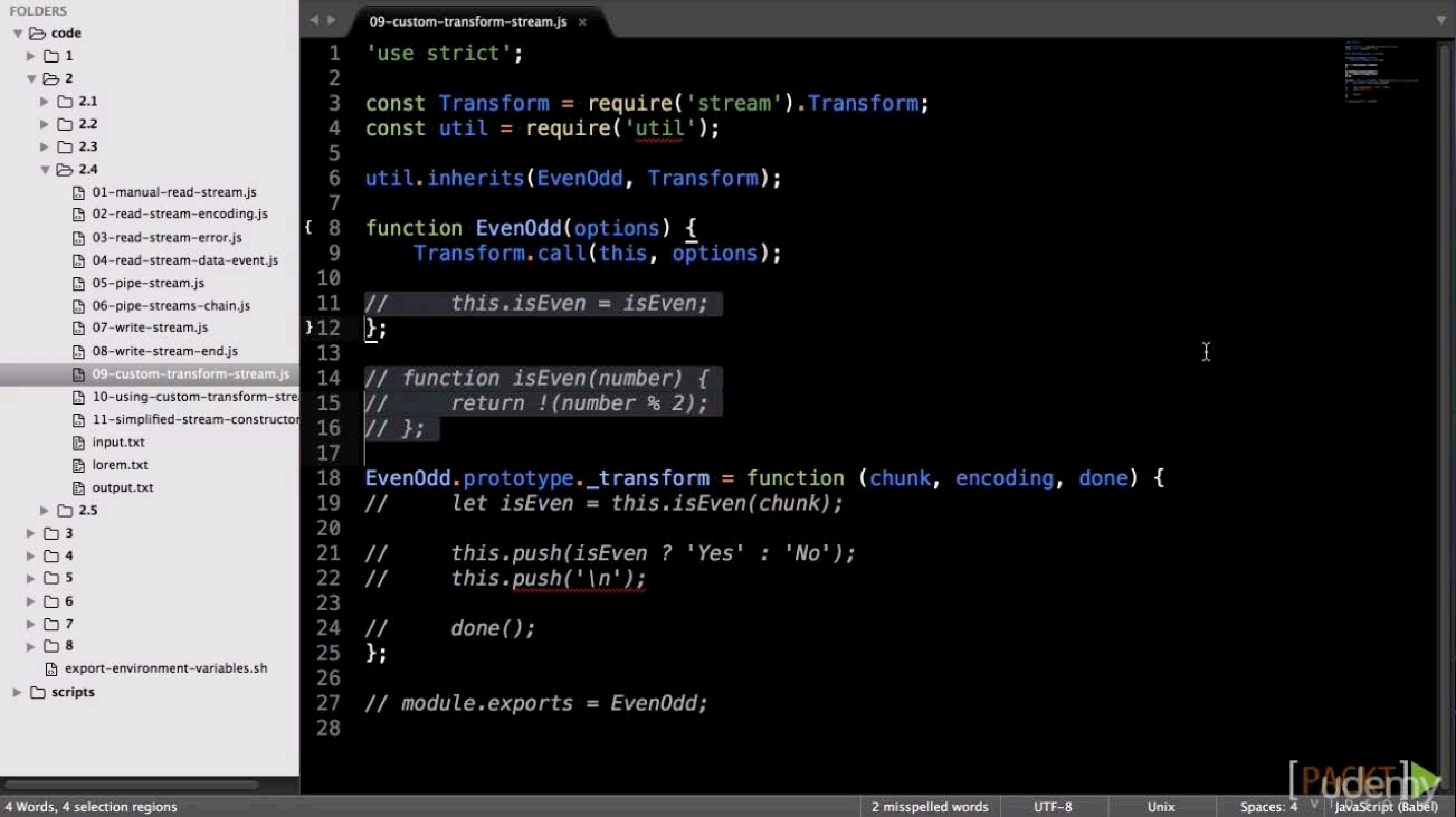
6 Words, 2 lines, 54 characters selected; Saved ~/projects/nodejs-mongodb-video-course/code/2/2.4/08-write-stream-end.js (UTF-8)



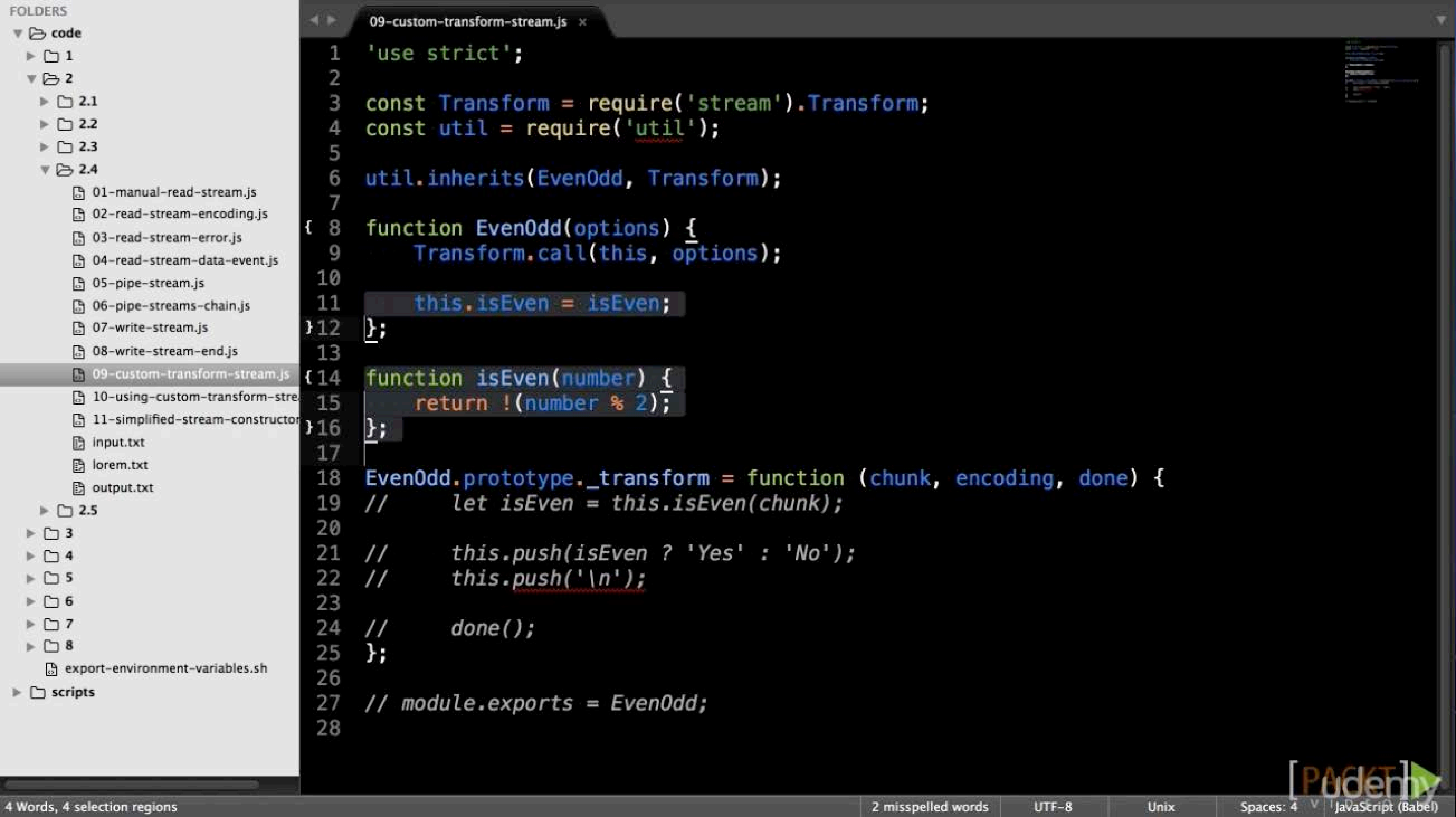
```
ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!]
$ node 08-write-stream-end.js
Did you try to write after calling end() method?
Error: write after end
    at writeAfterEnd (_stream_writable.js:160:12)
    at WriteStream.Writable.write (_stream_writable.js:205:5)
    at Object.<anonymous> (/Users/ifdattic/projects/nodejs-mongodb-video-course/code/2/2.4/08-write-stream-end.js:20:8)
    at Module._compile (module.js:399:26)
    at Object.Module._extensions..js (module.js:406:10)
    at Module.load (module.js:345:32)
    at Function.Module._load (module.js:302:12)
    at Function.Module.runMain (module.js:431:10)
    at startup (node.js:141:18)
    at node.js:977:3
Finished writing to stream

ifdattic at ifdattic-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!?]
$ 
```

Custom Streams

Use case	Class	Method(s) to implement
Reading only	Readable	_read
Writing only	Writable	_write, _writev
Reading and writing	Duplex	_read, _write, _writev
Operate on written data, then read the result	Transform	_transform, _flush



```
'use strict';

const Transform = require('stream').Transform;
const util = require('util');

util.inherits(EvenOdd, Transform);

function EvenOdd(options) {
    Transform.call(this, options);

//      this.isEven = isEven;
};

// function isEven(number) {
//      return !(number % 2);
// };

EvenOdd.prototype._transform = function (chunk, encoding, done) {
//      let isEven = this.isEven(chunk);

//      this.push(isEven ? 'Yes' : 'No');
//      this.push('\n');

//      done();
};

// module.exports = EvenOdd;
```

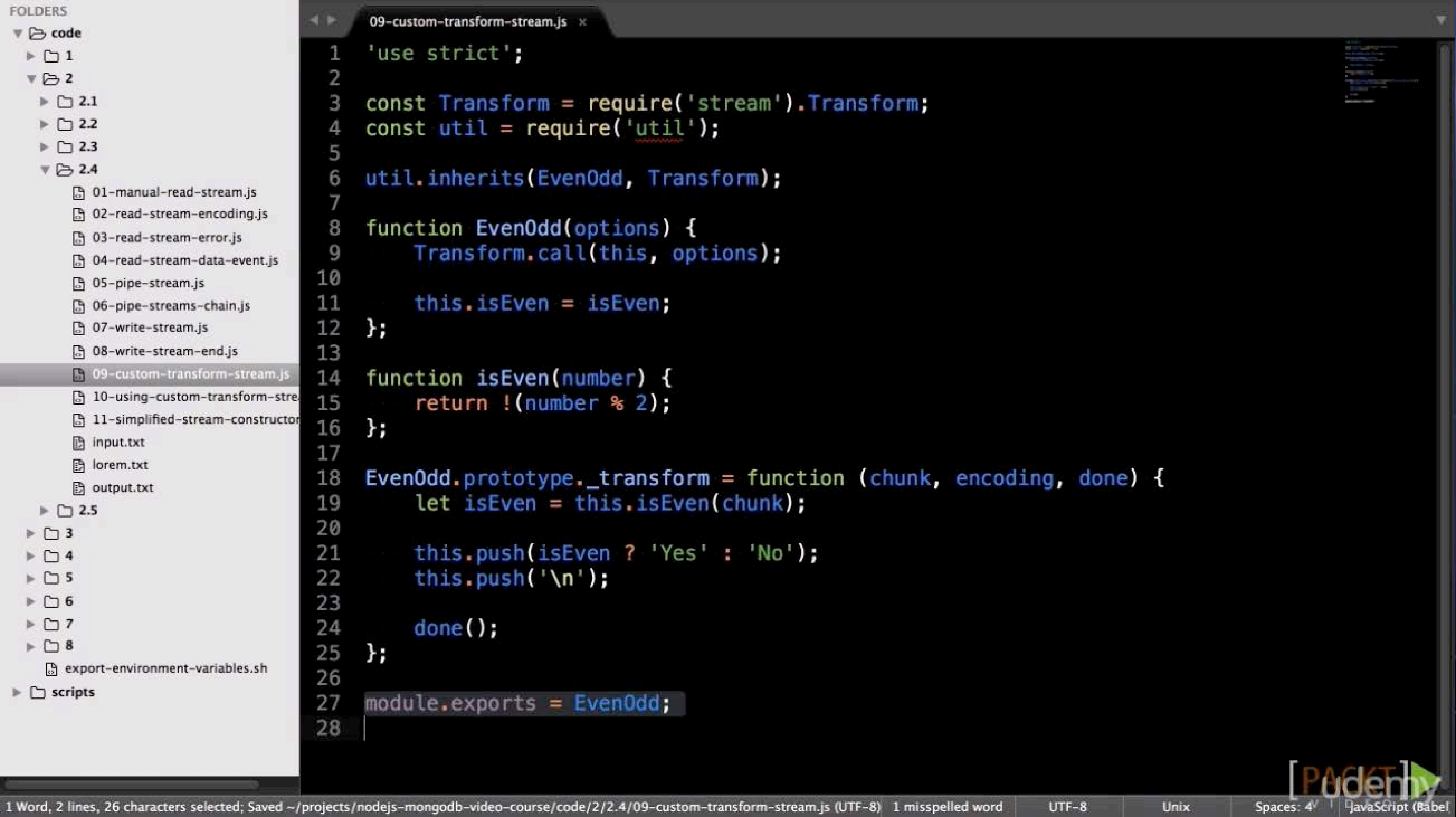

FOLDERS

- code
  - 1
  - 2
    - 2.1
    - 2.2
    - 2.3
    - 2.4
      - 01-manual-read-stream.js
      - 02-read-stream-encoding.js
      - 03-read-stream-error.js
      - 04-read-stream-data-event.js
      - 05-pipe-stream.js
      - 06-pipe-streams-chain.js
      - 07-write-stream.js
      - 08-write-stream-end.js
      - 09-custom-transform-stream.js
      - 10-using-custom-transform-stre
      - 11-simplified-stream-constructor
      - input.txt
      - lorem.txt
      - output.txt
    - 2.5
  - 3
  - 4
  - 5
  - 6
  - 7
  - 8
  - export-environment-variables.sh
- scripts

09-custom-transform-stream.js

```javascript
'use strict';

const Transform = require('stream').Transform;
const util = require('util');

util.inherits(EvenOdd, Transform);

function EvenOdd(options) {
    Transform.call(this, options);

    this.isEven = isEven;
};

function isEven(number) {
    return !(number % 2);
};

EvenOdd.prototype._transform = function (chunk, encoding, done) {
//     let isEven = this.isEven(chunk);

//     this.push(isEven ? 'Yes' : 'No');
//     this.push('\n');

//     done();
};

// module.exports = EvenOdd;

```

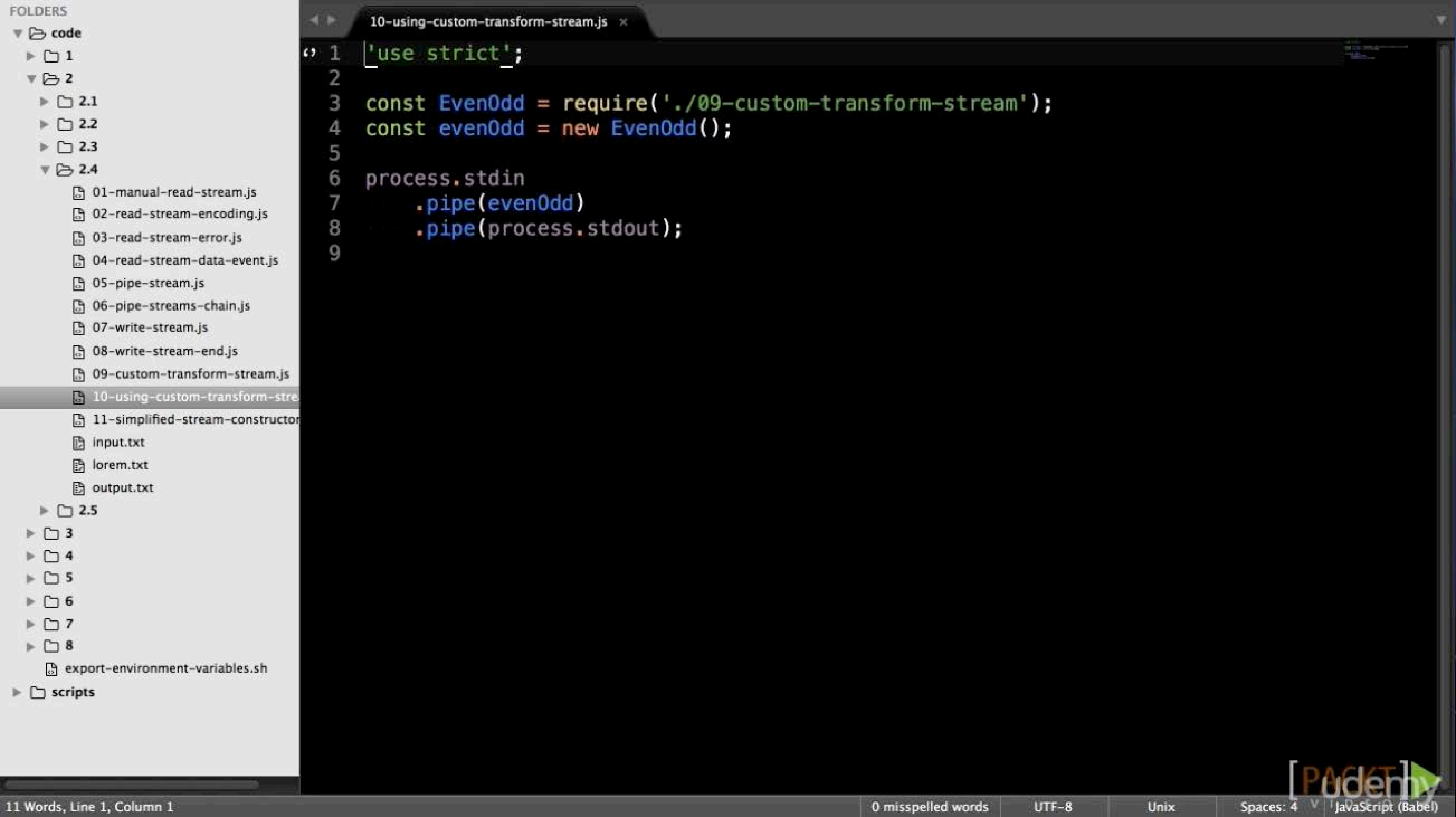
4 Words, 4 selection regions | 2 misspelled words | UTF-8 | Unix | Spaces: 4 | JavaScript (Babel)



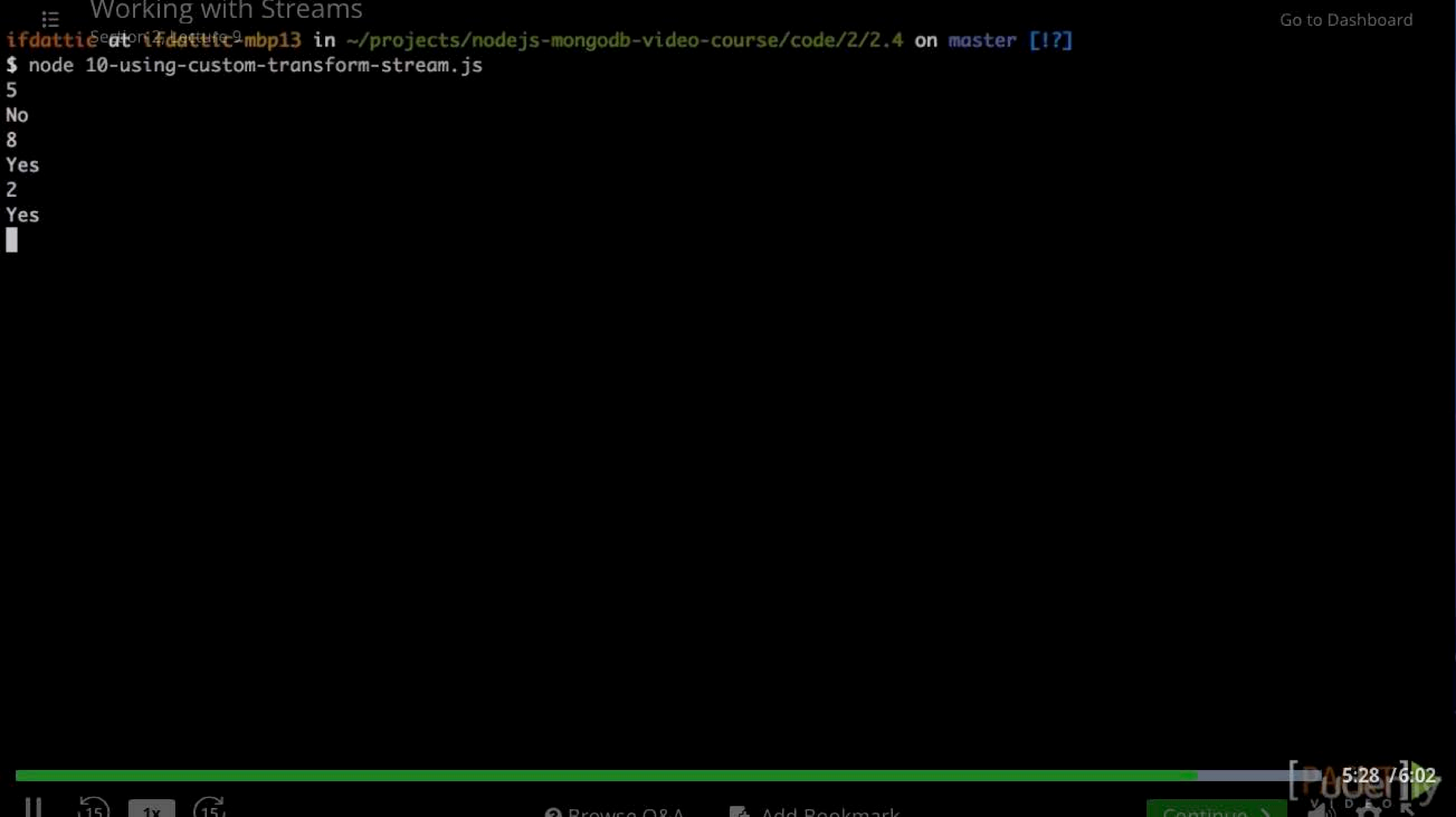
FOLDERS
code
1
2
2.1
2.2
2.3
2.4
01-manual-read-stream.js
02-read-stream-encoding.js
03-read-stream-error.js
04-read-stream-data-event.js
05-pipe-stream.js
06-pipe-streams-chain.js
07-write-stream.js
08-write-stream-end.js
09-custom-transform-stream.js
10-using-custom-transform-stre
11-simplified-stream-constructor
input.txt
lorem.txt
output.txt
2.5
3
4
5
6
7
8
export-environment-variables.sh
scripts
09-custom-transform-stream.js

```javascript
'use strict';

const Transform = require('stream').Transform;
const util = require('util');

util.inherits(EvenOdd, Transform);

function EvenOdd(options) {
    Transform.call(this, options);

    this.isEven = isEven;
};

function isEven(number) {
    return !(number % 2);
};

EvenOdd.prototype._transform = function (chunk, encoding, done) {
    let isEven = this.isEven(chunk);

    this.push(isEven ? 'Yes' : 'No');
    this.push('\n');

    done();
};

module.exports = EvenOdd;
```

```
1 Word, 2 lines, 26 characters selected; Saved ~/projects/nodejs-mongodb-video-course/code/2/2.4/09-custom-transform-stream.js (UTF-8)   1 misspelled word   UTF-8   Unix   Spaces: 4   JavaScript (Babel
```



```
'use strict';

const EvenOdd = require('./09-custom-transform-stream');
const evenOdd = new EvenOdd();

process.stdin
    .pipe(evenOdd)
    .pipe(process.stdout);
```

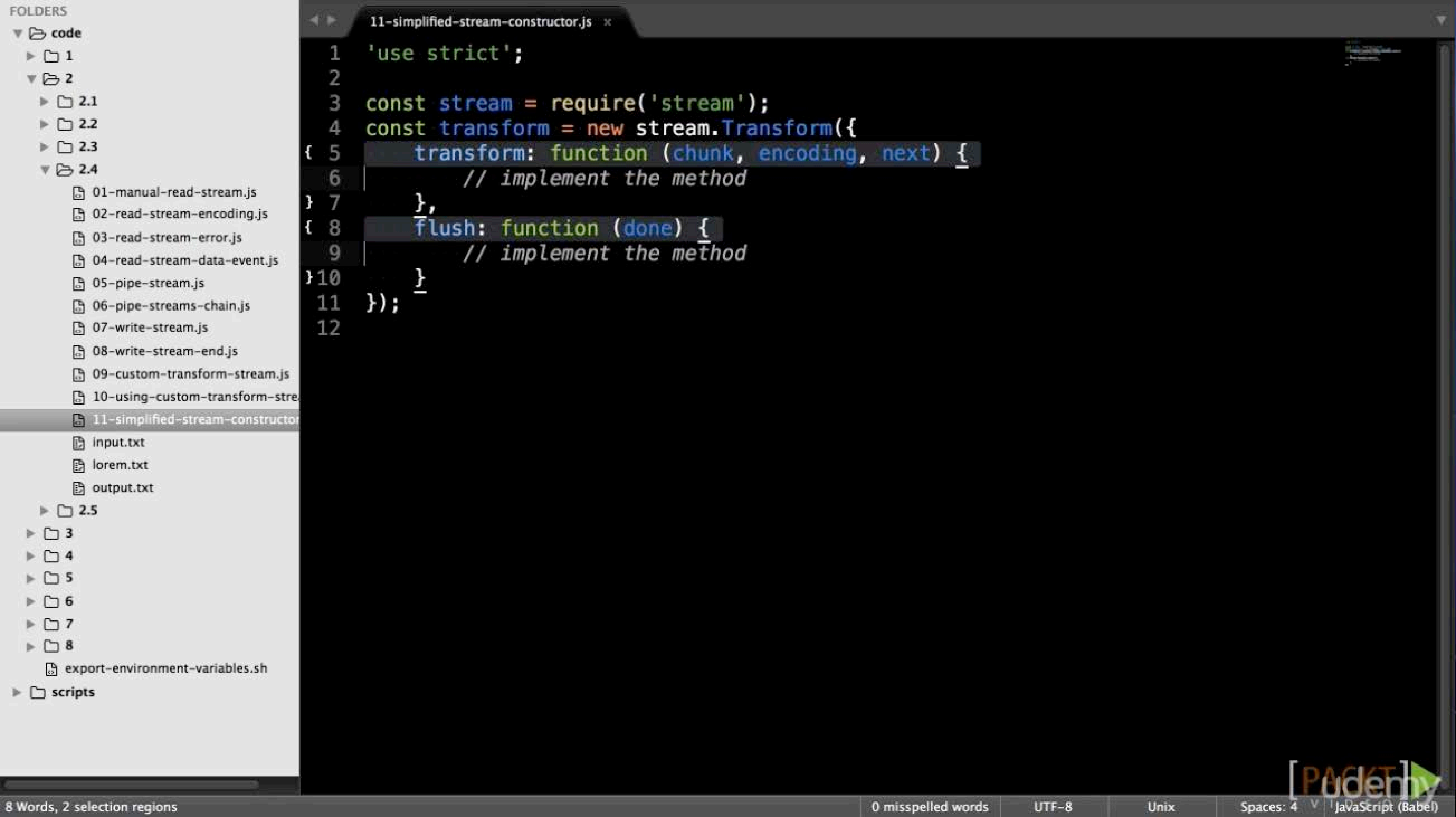


Working with Streams

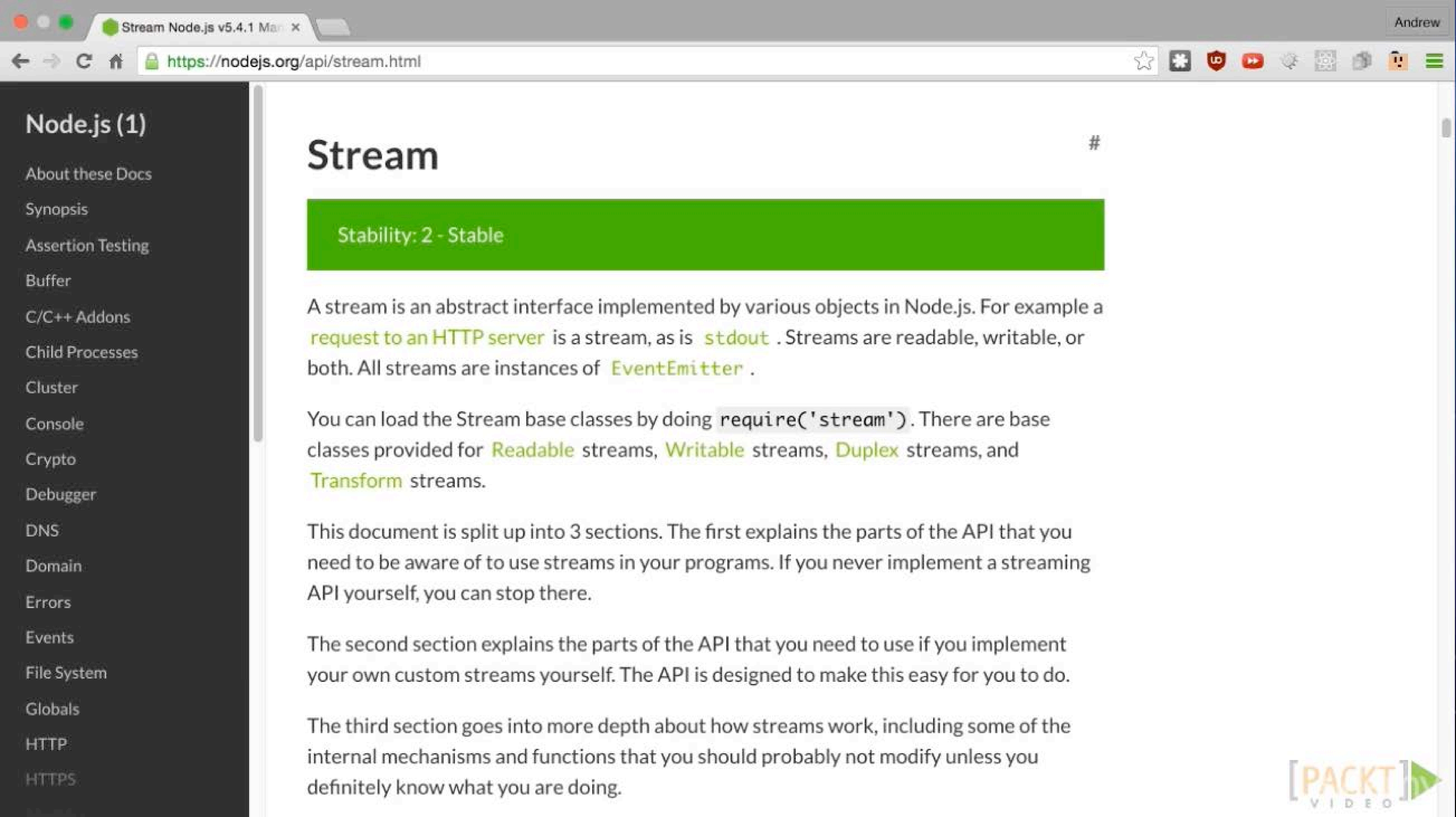
Section 2, Lecture 9

Go to Dashboard

```
ifdattie at ifdattie-mbp13 in ~/projects/nodejs-mongodb-video-course/code/2/2.4 on master [!?]
$ node 10-using-custom-transform-stream.js
5
No
8
Yes
2
Yes
```



```javascript
'use strict';

const stream = require('stream');
const transform = new stream.Transform({
    transform: function (chunk, encoding, next) {
        // implement the method
    },
    flush: function (done) {
        // implement the method
    }
});
```



Node.js (1)

About these Docs
Synopsis
Assertion Testing
Buffer
C/C++ Addons
Child Processes
Cluster
Console
Crypto
Debugger
DNS
Domain
Errors
Events
File System
Globals
HTTP
HTTPS

Stream

Stability: 2 - Stable

A stream is an abstract interface implemented by various objects in Node.js. For example a request to an HTTP server is a stream, as is `stdout`. Streams are readable, writable, or both. All streams are instances of `EventEmitter`.

You can load the Stream base classes by doing `require('stream')`. There are base classes provided for Readable streams, Writable streams, Duplex streams, and Transform streams.

This document is split up into 3 sections. The first explains the parts of the API that you need to be aware of to use streams in your programs. If you never implement a streaming API yourself, you can stop there.

The second section explains the parts of the API that you need to use if you implement your own custom streams yourself. The API is designed to make this easy for you to do.

The third section goes into more depth about how streams work, including some of the internal mechanisms and functions that you should probably not modify unless you definitely know what you are doing.

Next Video

Working with Files