Error: Can't set headers after they are sent to the client

I'm fairly new to Node.js and I am having some issues.

I am using Node.js 4.10 and Express 2.4.3.

When I try to access http://127.0.0.1:8888/auth/facebook, i'll be redirected to http://127.0.0.1:8888/auth/facebook_callback.

I then received the following error:

```
Error: Can't render headers after they are sent to the client.
    at ServerResponse.<anonymous> (http.js:573:11)
    at ServerResponse._renderHeaders
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:64:

    at ServerResponse.writeHead (http.js:813:20)
    at /home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/auth.strategies/facebook.js:28:15
    at /home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/index.js:113:13
    at next (/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/strategyExecutor.js:45:39)
    at [object Object].pass
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/authExecutionScope.js:32:3)
    at [object Object].halt
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/authExecutionScope.js:29:8)
    at [object Object].redirect
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/authExecutionScope.js:16:8)
    at [object Object].<anonymous>
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/auth.strategies/facebook.js:77:15)
Error: Can't set headers after they are sent.
    at ServerResponse.<anonymous> (http.js:527:11)
    at ServerResponse.setHeader
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:50:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:162:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:195:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:150:

    at param
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at pass
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at Object.router [as handle]
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:198:

    at Object.auth [as handle]
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/index.js:153:7)
Error: Can't set headers after they are sent.
    at ServerResponse.<anonymous> (http.js:527:11)
    at ServerResponse.setHeader
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:50:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:162:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:207:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:150:

    at param
```

```
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at pass
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at Object.router [as handle]
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/r

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:198:

    at Object.auth [as handle]
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/index.js:153:7)
Error: Can't set headers after they are sent.
    at ServerResponse.<anonymous> (http.js:527:11)
    at ServerResponse.setHeader
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:50:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:162:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:150:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:207:

    at Object.auth [as handle]
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect-
auth/lib/index.js:153:7)
    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:198:

    at HTTPServer.handle
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:211:

    at Object.handle
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:105:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:198:

Error: Can't set headers after they are sent.
    at ServerResponse.<anonymous> (http.js:527:11)
    at ServerResponse.setHeader
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:50:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:162:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:150:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:207:

    at HTTPServer.handle
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:211:

    at Object.handle
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:105:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:198:

    at
/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/se

    at
/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/se


node.js:134
        throw e; // process.nextTick error, or 'error' event on first tick
        ^
Error: Can't set headers after they are sent.
    at ServerResponse.<anonymous> (http.js:527:11)
    at ServerResponse.setHeader
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/patch.js:50:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:162:

    at next
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/http.js:207:

    at
/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/se

    at
/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/se

    at Array.<anonymous>
```

```
(/home/eugene/public_html/all_things_node/projects/fb2/node_modules/connect/lib/middleware/s

    at EventEmitter._tickCallback (node.js:126:26)
```

The following is my code:

```
var fbId= "XXX";
var fbSecret= "XXXXXX";
var fbCallbackAddress= "http://127.0.0.1:8888/auth/facebook_callback"

var cookieSecret = "node";     // enter a random hash for security

var express= require('express');
var auth = require('connect-auth')
var app = express.createServer();


app.configure(function(){
    app.use(express.bodyParser());
    app.use(express.methodOverride());
    app.use(express.cookieParser());
    app.use(express.session({secret: cookieSecret}));
    app.use(auth([
        auth.Facebook({
            appId : fbId,
            appSecret: fbSecret,
            callback: fbCallbackAddress,
            scope:
'offline_access,email,user_about_me,user_activities,manage_pages,publish_stream',
            failedUri: '/noauth'
        })
    ]));
    app.use(app.router);
});


app.get('/auth/facebook', function(req, res) {
  req.authenticate("facebook", function(error, authenticated) {
    if (authenticated) {
      res.redirect("/great");
      console.log("ok cool.");
      console.log(res['req']['session']);
    }
  });
});

app.get('/noauth', function(req, res) {
  console.log('Authentication Failed');
  res.send('Authentication Failed');
});

app.get('/great', function( req, res) {
  res.send('Supercoolstuff');
});

app.listen(8888);
```

May I know what is wrong with my code?

I'm really new to this, so sorry for just putting up the code here.

Thank you all in advance.

javascript node.js express

edited Jul 11 at 19:40
Praveen Kumar
114k 20 100 152

asked Aug 12 '11 at 15:21
DjangoRocks
2,550 6 26 46

protected by Community ♦ Nov 24 '15 at 16:00

This question is protected to prevent "thanks!", "me too!", or spam answers by new users. To answer it, you must have earned at least 10 reputation on this site (the association bonus does not count).

Google sent me to this question, but newer versions of ExpressJS have res.headersSent boolean which can be used to check if safe to set/send headers – Julian Soro Apr 28 at 22:53

14 Answers

The `res` object in Express is a subclass of Node.js's `http.ServerResponse` (read the http.js source). You are allowed to call `res.setHeader(name, value)` as often as you want until you call

`res.writeHead(statusCode)`. After `writeHead`, the headers are baked in and you can only call `res.write(data)`, and finally `res.end(data)`.

The error "Error: Can't set headers after they are sent." means that you're already in the Body or Finished state, but some function tried to set a header or statusCode. When you see this error, try to look for anything that tries to send a header after some of the body has already been written. For example, look for callbacks that are accidentally called twice, or any error that happens after the body is sent.

In your case, you called `res.redirect()`, which caused the response to become Finished. Then your code threw an error (`res.req` is `null`). and since the error happened within your actual `function(req, res, next)` (not within a callback), Connect was able to catch it and then tried to send a 500 error page. But since the headers were already sent, Node.js's `setHeader` threw the error that you saw.

Comprehensive list of Node.js/Express response methods and when they must be called:

Response must be in **Head** and remains in **Head**:

1. `res.writeContinue()`
2. `res.statusCode = 404`
3. `res.setHeader(name, value)`
4. `res.getHeader(name)`
5. `res.removeHeader(name)`
6. `res.header(key[, val])` (Express only)
7. `res.charset = 'utf-8'` (Express only; only affects Express-specific methods)
8. `res.contentType(type)` (Express only)

Response must be in **Head** and becomes **Body**:

1. `res.writeHead(statusCode, [reasonPhrase], [headers])`

Response can be in either **Head/Body** and remains in **Body**:

1. `res.write(chunk, encoding='utf8')`

Response can be in either **Head/Body** and becomes **Finished**:

1. `res.end([data], [encoding])`

Response can be in either **Head/Body** and remains in its current state:

1. `res.addTrailers(headers)`

Response must be in **Head** and becomes **Finished**:

1. `return next([err])` (Connect/Express only)
2. Any exceptions within middleware `function(req, res, next)` (Connect/Express only)
3. `res.send(body|status[, headers|status[, status]])` (Express only)
4. `res.attachment(filename)` (Express only)
5. `res.sendfile(path[, options[, callback]])` (Express only)
6. `res.json(obj[, headers|status[, status]])` (Express only)
7. `res.redirect(url[, status])` (Express only)
8. `res.cookie(name, val[, options])` (Express only)
9. `res.clearCookie(name[, options])` (Express only)
10. `res.render(view[, options[, fn]])` (Express only)
11. `res.partial(view[, options])` (Express only)

edited Nov 18 '15 at 1:29

community wiki
3 revs, 2 users 99%
yonran

32 callback twice was my issue - thx. – bryanmac Jun 28 '13 at 14:08

1 No more banging my head. Thank you! – Adam Grant Jul 30 '14 at 19:02

2 Yep, check for calling next() or other cb twice. – Tony Gutierrez May 6 '15 at 18:19

5 also watch out for this classic mistake: res.redirect() doesn't stop statement execution... so return after it. Otherwise other code could be executed which could unintentiallly cause the famous header error. Thanx for the explanation! – KLoozen Jan 23 '16 at 11:44

1 I made a very small error in my middleware, I didnt `return` before `next()` , thanks this pointed me to the error! – illcrx Dec 30 '16 at 20:32

I ran into this error as well for a while. I think (hope) I've wrapped my head around it, wanted to write it here for reference.

When you add middleware to connect or express (which is built on connect) using the `app.use` method, you're appending items to `Server.prototype.stack` in connect (At least with the current `npm install connect` , which looks quite different from the one github as of this post). When the server gets a request, it iterates over the stack, calling the `(request, response, next)` method.

The problem is, if in one of the middleware items writes to the response body or headers (it looks like it's either/or for some reason), **but doesn't call `response.end()` and you call `next()`** then as the core `Server.prototype.handle` method completes, it's going to notice that:

1. there are no more items in the stack, and/or
2. that `response.headerSent` is true.

So, it throws an error. But the error it throws is just this basic response (from the connect `http.js` source code:

```
res.statusCode = 404;
res.setHeader('Content-Type', 'text/plain');
res.end('Cannot ' + req.method + ' ' + req.url);
```

Right there, it's calling `res.setHeader('Content-Type', 'text/plain');` , which you are likely to have set in your `render` method, **without calling response.end()**, something like:

```
response.setHeader("Content-Type", "text/html");
response.write("<p>Hello World</p>");
```

The way everything needs to be structured is like this:

Good Middleware

```
// middleware that does not modify the response body
var doesNotModifyBody = function(request, response, next) {
  request.params = {
    a: "b"
  };
  // calls next because it hasn't modified the header
  next();
};

// middleware that modify the response body
var doesModifyBody = function(request, response, next) {
  response.setHeader("Content-Type", "text/html");
  response.write("<p>Hello World</p>");
  response.end();
  // doesn't call next()
};

app.use(doesNotModifyBody);
app.use(doesModifyBody);
```

Problematic Middleware

```
var problemMiddleware = function(request, response, next) {
  response.setHeader("Content-Type", "text/html");
  response.write("<p>Hello World</p>");
  next();
};
```

The problematic middleware sets the response header without calling `response.end()` and calls `next()` , which confuses connect's server.

answered Oct 17 '11 at 3:38

Lance Pollard
27.3k 58 181 309

3 +1 This is a great explanation, but what about the case when you use res.redirect()? I frequently run into this problem when the middleware is trying to redirect based on some condition. Should middleware not redirect, per your "Good Middleware" example? – qodeninja Dec 6 '12 at 22:45

You know I have this exact problem due to what you call a problematic middleware, however I need a case where I return response but would like to do further processing in a separate controller as part of the chain, how do I go about suppressing this error? – iQ. Oct 30 '14 at 14:03

I had this same issue and realised it was because I was calling res.redirect without a return statement, so the next function was also being called immediately afterwards:

```
auth.annonymousOnly = function(req, res, next) {
    if (req.user) res.redirect('/');
    next();
};
```

Which should have been:

```
auth.annonymousOnly = function(req, res, next) {
    if (req.user) return res.redirect('/');
    next();
};
```

edited Feb 15 '15 at 10:00

answered Feb 14 '15 at 13:02

ergusto
355 4 10

1 THANK YOU !! That was it. – Atrahasis Apr 11 '16 at 13:29

1 Yup, that was the same issue I had. Thanks – Recap Dec 5 '16 at 16:27

Lots of people hit this error. It's a confusing this with async processing. Most likely some of your code is setting headers in the first tick and then you are running an async callback in a future tick. In between, the response header gets sent, but then further headers (like a 30X redirect) try to add extra headers, but it's too late since the response header has already been transmitted.

I'm not sure exactly what's causing your error, but look at any callbacks as potential areas to investigate.

One easy tip to simplify your code. Get rid of app.configure() and just call app.use directly in your top level scope.

See also the everyauth module, which does Facebook and a dozen or so other 3rd party authentication providers.

edited Nov 3 '15 at 9:54

Alex Booker
4,543 1 8 26

answered Aug 12 '11 at 18:24

Peter Lyons
93.4k 20 183 203

hello, thanks! will look into it. and get back to you. – DjangoRocks Aug 13 '11 at 6:44

What is a `30X redirect` ? – Randomblue Dec 23 '11 at 15:30

A 30X redirect is an HTTP response code. w3.org/Protocols/rfc2616/rfc2616-sec10.html Codes 300-399 are different variations of redirection, with 302 and 301 being commonly used to send the client to an alternate URL. When you do response.redirect(...) in node, a 30X redirect header will be sent in the response. – Peter Lyons Dec 23 '11 at 21:23

I boiled my head over this issue and it has happened due to a careless mistake on handling the callbacks. non returned callbacks cause the response to be set twice.!

My program had a code which validate request and query the DB. after validating if error is there, I was calling back the index.js with the validation errors . And if validation passes it goes ahead and hit the db with success/failure.

```
    var error = validateRequestDetails("create",queryReq);
    if (error)
        callback(error, null);
   else
```

```
    some code
    callback(null, success);
```

What was happening is : Incase validation fails the callback get called and response get set. But not returned. So it still continues the method goes to db and hit success/failure . It calls the same callback again causing the response to be set twice now.

So solution is simple, you need to 'return' the callback so that the method don't continue executing, once the error has occurred and hence set the response object once

```
var error = validateRequestDetails("create",queryReq);
  if (error)
      callback(error, null);
      return;
  else
     some code
     callback(null, success);
```

edited Aug 31 '15 at 5:13

answered Jul 4 '15 at 5:22

randomness
643 8 15

Thanks! This turned out to be my problem too. Just did a ctrl+f and found a `callback(...)` without a `return;` after it which was eventually causing `res.send(...)` to be called twice. – JoeRocc Jun 13 '16 at 16:22

This type of error you will get when you pass statements after sending a response.

For example:

```
res.send("something response");
console.log("jhgfjhgsdhgfsdf");
console.log("sdgsdfhdgfdhgsdf");
res.send("sopmething response");
```

Will result in the error you are seeing, because once the response has been sent, the following `res.send` will not be executed.

If you want do anything, you should do it before sending the response.

edited Feb 27 at 15:47

dKen
2,056 1 14 27

answered Sep 28 '16 at 13:21

Trojan
358 4 8

In my case it was a 304 response (caching) that was causing the issue.

Easiest solution:

```
app.disable('etag');
```

Alternate solution here if you want more control:

http://vlasenko.org/2011/10/12/expressconnect-static-set-last-modified-to-now-to-avoid-304-not-modified/

answered May 17 '14 at 0:05

blented
1,066 13 14

In my case also 304 response. I a m using Fibers for processing. Any way your answer helps a lot. thank you – Dileep stanley Jul 26 '14 at 16:00

Can anyone explain what the implications are for removing the etag header? – mattwilsn May 23 '16 at 16:17

1 ETags allow the server to not send content that hasn't changed. Turning it off disables this feature. The ETag wikipedia entry (en.wikipedia.org/wiki/HTTP_ETag) has a lengthier explanation. – blented May 24 '16 at 18:25

In my case this happened with React and postal.js when I didn't unsubscribe from a channel in the `componentWillUnmount` callback of my React component.

answered Jun 2 '15 at 10:12

Zoltán


10.4k 7 49 91

For anyone that's coming to this and none of the other solutions helped, in my case this manifested on a route that handled image uploading but didn't handle ***timeouts***, and thus if the upload took too long and timed out, when the callback was fired *after the timeout response had been sent*, calling res.send() resulted in the crash as the headers were already set to account for the timeout.

This was easily reproduced by setting a very short timeout and hitting the route with a decently-large image, the crash was reproduced every time.

answered May 19 '15 at 12:56


Mike
6,001 3 20 54

Just leaned this. You can pass the responses through this function:

```
app.use(function(req,res,next){
  var _send = res.send;
  var sent = false;
  res.send = function(data){
    if(sent) return;
    _send.bind(res)(data);
    sent = true;
};
  next();
});
```

answered Nov 5 '15 at 7:28


iMad
1,176 1 15 24

Some times you may get this error when you try to call next() function after **res.end** or **res.send** , try to delete if you have next() after res.send or res.end in your function.

ex :

```
router.get('/',function (req,res,next){
     res.send("request received");
     next(); ---> this will give you the above exception
});
```

remove next() from above function and it will work.

answered Mar 8 at 11:54


Surendra cool
23 6

This happens when response was delivered to client and again you are trying to give response. You have to check in your code that somewhere you are returning response to client again which causes this error. Check and return response once when you want to return.

answered Jul 12 at 13:45


Ankit Manchanda
78 6

Add this middlware and it will work

```
app.use(function(req,res,next){
 var _send = res.send;
var sent = false;
res.send = function(data){
    if(sent) return;
    _send.bind(res)(data);
    sent = true;
};
next();
});
```

answered Apr 28 at 4:26

ASHISH RANJAN


89 3

simple answer from visionmedia:
https://github.com/visionmedia/express/issues/634

answered Nov 30 '11 at 4:34


shi11i
1,181 9 13