Javascript ES6 TypeError: Class constructor Client cannot be invoked without 'new'

Asked 6 years, 4 months ago Modified 9 months ago Viewed 250k times

84

I have a class written in Javascript ES6. When I try to execute `nodemon` command I always see this error `TypeError: Class constructor Client cannot be invoked without 'new'`

The full error is mentioned below:

```
/Users/akshaysood/Blockchain/fabricSDK/dist/application/Transaction.js:45
        return (0, _possibleConstructorReturn3.default)(this,
(FBClient.__proto__ || (0, _getPrototypeOf2.default)(FBClient)).call(this,
props));

^

TypeError: Class constructor Client cannot be invoked without 'new'
    at new FBClient
(/Users/akshaysood/Blockchain/fabricSDK/dist/application/Transaction.js:45:127)
    at Object.<anonymous>
(/Users/akshaysood/Blockchain/fabricSDK/dist/application/Transaction.js:195:14)
    at Module._compile (module.js:641:30)
    at Object.Module._extensions..js (module.js:652:10)
    at Module.load (module.js:560:32)
    at tryModuleLoad (module.js:503:12)
    at Function.Module._load (module.js:495:3)
    at Module.require (module.js:585:17)
    at require (internal/module.js:11:18)
    at Object.<anonymous>
(/Users/akshaysood/Blockchain/fabricSDK/dist/routes/users.js:11:20)
```

What I am trying to do is, I have created a class and then created an instance of that class. Then I am trying to export that variable.

The class structure is defined below:

```
class FBClient extends FabricClient{

    constructor(props){
        super(props);
    }

<<< FUNCTIONS >>>

}
```

How I am trying to export the variable ->

```
var client = new FBClient();
client.loadFromConfig(config);

export default client = client;
```

You can find the full code here --> Link. Code generated by Babel --> Link.

javascript node.js ecmascript-6 syntax-error

Share

Improve this question

Follow

edited Oct 20, 2022 at 16:19

Alexander Solonik

10.2k ● 19 ● 83 ● 184

asked Aug 15, 2018 at 13:51

Akshay Sood

6,756 ● 10 ● 41 ● 66

I'm not sure that you can export the instance of a class. I think you have to export the class, import it in another, then create the instance there – bwalshy Aug 15, 2018 at 13:54

8 @bwalshy - You can export any value, including an instance of a class. – T.J. Crowder Aug 15, 2018 at 13:55

@T.J.Crowder This code can not be run online as it requires Blockchain module. – Akshay Sood Aug 15, 2018 at 13:58

2 @AkshaySood - The error is clearly not about blockchain stuff, it's about basic class definition and instantiation. Please read that MCVE link. You're not expected to post all your code. You're expected to work through a process producing a minimal replicating example. You haven't even shown us the class where the error is occurring. – T.J. Crowder Aug 15, 2018 at 14:03

2 Are you using Babel to transpile your code? (possibly related issue, but just a guess) – Pointy Aug 15, 2018 at 14:04

11 Answers

Sorted by: Highest score (default)

121

The problem is that the class extends native ES6 class and is transpiled to ES5 with Babel. Transpiled classes cannot extend native classes, at least without additional measures.

```
class TranspiledFoo extends NativeBar {
  constructor() {
    super();
  }
}
```

results in something like

```
function TranspiledFoo() {
  var _this = NativeBar.call(this) || this;
  return _this;
}
// prototypically inherit from NativeBar
```

Since ES6 classes should be only called with `new`, `NativeBar.call` results in error.

ES6 classes are supported in any recent Node version, they shouldn't be transpiled. `es2015` should be excluded from Babel configuration, it's preferable to use `env` preset set to `node` target.

The same problem applies to TypeScript. The compiler should be properly configured to not transpile classes in order for them to inherit from native or Babel classes.

Share

Improve this answer

Follow

edited May 19, 2020 at 11:57

answered Aug 15, 2018 at 14:38

Estus Flask

221k ● 77 ● 468 ● 604

3 As the answer says. Change Babel configuration to not transpile to ES5. `es2015` preset is responsible for this behaviour. – Estus Flask Aug 15, 2018 at 14:45

If I do not transpile ES6 into ES5 using Babel. It throws error :unexpected token import: – Akshay Sood Aug 15, 2018 at 14:46

4 The manual covers that, babeljs.io/docs/en/babel-preset-env . Use 'target node'. – Estus Flask Aug 15, 2018 at 14:50

1 @jchook Technically it works like `function A() { if (!new.target) throw new Error('cannot be invoked without new') }`. `A` has its terms but it's definitely a function. – Estus Flask Feb 21, 2020 at 19:34

2 @Haroldo_OK Indeed, at least if this means the support of legacy browsers (browser app shouldn't necessarily be ES5 in 2020). If there's a class that extends built-in class (Error, Promise, etc.), it should be extended in a special way to be compatible with ES5, this is covered in other questions on SO. If a class extends native ES6 class, it's the existence of untranspiled class in ES5 app that is a real issue. – Estus Flask Feb 27, 2020 at 16:40

39

I was transpiling not Javascript but Typescript, and ran into the same problem.

I edited the Typescript compiler config file, `tsconfig.json`, to generate ES2017 Javascript code:

```
{
    "compilerOptions": {
        "target": "ES2017",
```

instead of whatever the default is, ES2015? — then all worked fine.

(Maybe this answer can be helpful for people who use Typescript and find this question when they search for the same error message, like I did.)

Share Improve this answer Follow

answered Apr 5, 2020 at 2:04

KajMagnus
11.6k • 17 • 85 • 132

This worked for me as well. changed "target": "es5" to >> "target": "es2017"
– Nadeesha Nawagaththegama Oct 29, 2023 at 15:06

this answer worked for me as well. i changed `"target": "es5"` to `"target": "ES2020"`
– Shemang David Jul 17 at 15:17

17

For those who are using ts-node, it could be possible that your `tsconfig.json` is unable to be loaded by **ts-node**.

1. Make sure you've set the below option for `tsconfig.json`:

```
{
    "compilerOptions": {
        "target": "ES6",
        ...
    }
}
```

2. Try `ts-node --script-mode` or use `--project` to specify the path of your `tsconfig.json`.

Share
Improve this answer
Follow

edited Mar 22, 2021 at 15:24

answered May 28, 2020 at 9:43

Devo
1,094 • 12 • 10

8

In `package.json` you can use targets configuration with `@babel/preset-env`. set the `esmodules` as 'true'.

Below is the example how I am using in my file:

```
"babel": {
  "presets": [
    [
      "@babel/preset-env",
      {
        "targets": {
          "esmodules": true
```

```
          }
        }
      ],
      "@babel/preset-react",
      "@babel/preset-flow"
    ],
    "plugins": [
      "@babel/plugin-proposal-class-properties"
    ]
  },
```

Share

Improve this answer

Follow

edited Apr 2, 2020 at 11:29

Hakan Dilek

2,237 ● 2 ● 24 ● 36

answered Apr 2, 2020 at 11:11

Chandan Aswal

89 ● 1 ● 2

3

For Angular 9+ this can be probably a matter of using typescript compilation flags also:

- `downlevelIteration`
- `importHelpers`

More info hereL https://www.typescriptlang.org/tsconfig#downlevelIteration.

Try to use this flags in your `tsconfig.josn`:

```
{
    "compilerOptions": {
        ...
        "downlevelIteration": true,
        "importHelpers": true,
        ...
    }
}
```

Without `"importHelpers": true` it should also works.

> This solution is mostly for Angular 9+ and ngcc compiler - this was an issue that occur for Angular 11. A custom configuration (custom ng-packgr config - Angular CLI library generating was introduced in Angular 6) for packages was maintained since there was Angular 4+ in the project. After miagration these packages to Angular 9+ libraries (projects folder) the error started occuring. I found that these flags (sctrictly downlevelIteration) solve exactly the problem.

Share

Improve this answer

edited Jul 15, 2021 at 10:56

answered Jul 14, 2021 at 19:28

TwaPaw

Follow 31 • 2

And also target to es6 – Xalion Apr 1, 2022 at 10:13

1

If you are not using babel and simple typescript. I got the error in material. Try to reduce the versions and bring them down to similar versions. My packages were having variety of versions.I solved the issue by lowering the package versions of packages like cdk, material.

```
"dependencies": {
  "@angular/animations": "~9.1.11",
  "@angular/cdk": "^9.2.4",
  "@angular/common": "~9.1.11",
  "@angular/compiler": "~9.1.11",
  "@angular/core": "~9.1.11",
  "@angular/forms": "~9.1.11",
  "@angular/material": "^9.2.4",
  "@angular/platform-browser": "~9.1.11",
  "@angular/platform-browser-dynamic": "~9.1.11",
  "@angular/router": "~9.1.11",
  "platform": "^1.3.5",
  "rxjs": "~6.5.4",
  "tslib": "^1.10.0",
  "zone.js": "~0.10.2"
}
```

Share
Improve this answer
Follow

edited Jul 4, 2020 at 1:18
jizhihaoSAMA
12.6k • 9 • 30 • 51

answered Jul 3, 2020 at 20:45
user3156438
11 • 1

1

Check your "entities" array in appmodule.ts I also missed that. When I kept the entity name that is used in query, issue resolved. Hope this answer helps.

Share Improve this answer Follow

answered Jan 30, 2023 at 17:47
Mohan Krishna
11 • 1

1

i used wrote before my ObjectId in nextjs and it worked.

Share Improve this answer Follow

answered Jul 5, 2023 at 18:01
P
Prakhar Singhal

11 ● 2

1 Your answer could be improved with additional supporting information. Please edit to add further details, such as citations or documentation, so that others can confirm that your answer is correct. You can find more information on how to write good answers in the help center. – Community Bot Jul 7, 2023 at 20:35

I wrote new before ObjectId in my Mongodb insertMany(), and it removes my error – Shawinder Jit Singh Feb 23 at 13:27

0

If you have already all these configurations on tsconfig and still it doesn't work maybe you have tsconfig.base file. try to chenge it too.

Share Improve this answer Follow

answered Feb 15 at 13:08

Tammar

25 ● 8

0

You can wrap instead of extending if you're not in TypeScript

In my situation, I didn't want to change the babel configuration (yet) and wanted to make the logic just work. Understanding a transpiled class cannot extend a native ES6 class I decided to simply wrap the base class, add functions to it and expose it (shoulnd't work in TypeScript).

My original code was something like:

```
import BaseClass from 'other-library';
class MyClass extends BaseClass {
  constructor(options) {
    super(options);

    this.on('event', this.action);
  }

  action(args) {
    // Custom logic for extending the BaseClass
  }
  addedAction(args) {
    // Custom logic for extending the BaseClass
  }
};
```

```
new MyClass({ args: ... }); // This throws "Class constructor BaseClass cannot
be invoked without 'new'"
```

And I changed inheritance to wrapping like this:

```
import BaseClass from 'other-library';
class MyClass {
  constructor(options) {
    this.wrapped = new BaseClass(options);
    this.wrapped.addedAction = this.addedAction.bind(this);
    this.wrapped.on('event', this.action);
  }

  action(args) {
    // Custom logic for extending the BaseClass
  }
  addedAction(args) {
    this.wrapped.doSomething();
  }
};

// To get the base instance:
new MyClass({ args: ... }).wrapped;
```

Notice that after creating an instance of `MyClass` I simply access `.wrapped` to get the `BaseClass` instance

Share Improve this answer Follow

answered Mar 24 at 11:00

Adam Tal

5,961 ● 4 ● 31 ● 49

-10

I hope you have already been able to solve your problem, here is my solution for this error:

```
class indexRoutes
{

    public  router: Router= Router();
}
const INDEX_ROUTES = new indexRoutes();
export default INDEX_ROUTES.router;
```

Share

Improve this answer

Follow

edited Aug 1, 2022 at 14:11

Tyler2P

2,370 ● 30 ● 25 ● 32

answered Oct 21, 2019 at 5:27

Alejandro

81 ● 1 ● 2