“Uncaught SyntaxError: Cannot use import statement outside a module” when importing ECMAScript 6

Asked 1 year, 8 months ago Active yesterday Viewed 1.1m times

391

50

I'm using ArcGIS JSAPI 4.12 and wish to use Spatial Illusions to draw military symbols on a map.

When I add `milsymbol.js` to the script, the console returns error

> Uncaught SyntaxError: Cannot use import statement outside a module`

so I add `type="module"` to the script, and then it returns

> Uncaught ReferenceError: ms is not defined

Here's my code:

```
<link rel="stylesheet" href="https://js.arcgis.com/4.12/esri/css/main.css">
<script src="https://js.arcgis.com/4.12/"></script>
<script type="module" src="milsymbol-2.0.0/src/milsymbol.js"></script>

<script>
    require([
        "esri/Map",
        "esri/views/MapView",
        "esri/layers/MapImageLayer",
        "esri/layers/FeatureLayer"
    ], function (Map, MapView, MapImageLayer, FeatureLayer) {

        var symbol = new ms.Symbol("SFG-UCI----D", { size: 30 }).asCanvas(3);
        var map = new Map({
            basemap: "topo-vector"
        });

        var view = new MapView({
            container: "viewDiv",
            map: map,
            center: [121, 23],
            zoom: 7
        });
    });
</script>
```

So, whether I add `type="module"` or not, there are always errors. However, in the official document of Spatial Illusions, there isn't any `type="module"` in the script. I'm now really confused. How do they manage to get it work without adding the type?

File *milsymbol.js*

Share Improve this question Follow

edited Mar 8 '20 at 22:36

Peter Mortensen
28.4k 21 95 123

asked Oct 3 '19 at 3:20

Jerry Chen
4,369 2 9 13

3 I am getting the same error while trying to import a module! Do you get any solution? – Zeeshan Ahmad Khalil Oct 29 '19 at 5:17

2 I am now using browserify through which i can include any module by using `require()` . Check out this video – Zeeshan Ahmad Khalil Oct 29 '19 at 6:54

18 Answers

Active | Oldest | Votes

240

I got this error because I forgot the type="module" inside the script tag:

```
<script type="module" src="milsymbol-2.0.0/src/milsymbol.js"></script>
```

Share Improve this answer Follow

answered Nov 3 '19 at 10:43

Emmanuel
3,906 2 19 22

136

+50

It looks like the cause of the errors are:

1. You're currently loading the source file in the `src` directory instead of the built file in the `dist` directory (you can see what the intended distributed file is here). This means that you're using the native source code in an unaltered/unbundled state, leading to the following error: `Uncaught SyntaxError: Cannot use import statement outside a module` . This should be fixed by using the bundled version since the package is using rollup to create a bundle.
2. The reason you're getting the `Uncaught ReferenceError: ms is not defined` error is because modules are scoped, and since you're loading the library using native modules, `ms` is not in the global scope and is therefore not accessible in the following script tag.

It looks like you should be able to load the `dist` version of this file to have `ms` defined on the `window` . Check out this example from the library author to see an example of how this can be done.

Share Improve this answer Follow

edited Jun 25 '20 at 0:19

I Support The Boycott
65k 39 209 352

answered Oct 3 '19 at 4:31

Kai
1,805 1 9 7

Thank you for your reply, now I know I have the wrong file. I've been looking for the dist version of the file but with no result. Do you know any way to get the dist version? Thanks so much! – Jerry Chen Oct 3 '19 at 5:58

defined in `src/milsymbol.js` , but it requires `type="module"` and will cause scope problem. Is there any solution for this. Thanks so much! – Jerry Chen Oct 3 '19 at 6:33

What if thats the actual intention, referencing it from /src. As the author is not planning to expose a property of a class for example.. – Cristian E. Jan 18 at 17:20

67

Update For Node / NPM

Add `"type": "module"` to your `package.json`

```
{
  // ...
  "type": "module",
  // ...
}
```

Note: When using modules, if you get `ReferenceError: require is not defined` , you'll need to use the `import` syntax instead of `require` . You can't natively mix and match between them, so you'll need to pick one or use a bundler if you need to use both.

Share Improve this answer Follow

edited Mar 1 at 18:04

answered Nov 3 '20 at 0:27

KyleMit
43k 54 369 544

33 I'm coming from this answer and now I'm in an infinite loop – wormsparty Mar 9 at 12:23

Did you found any solution @wormsparty – Deepak Kumar Mar 13 at 9:33

Yes. Basically, don't run TypeScript scripts independently, but put them in an existing Angular project and things will work fine ;-) – wormsparty Mar 14 at 10:03

@wormsparty Escape the loop – tejasvi88 May 15 at 15:18

43

I was also facing the same issue until I added the type="module" to the script.

Before it was like this

```
<script src="../src/main.js"></script>
```

And after changing it to

```
<script type="module" src="../src/main.js"></script>
```

It worked perfectly.

US/docs/Web/HTTP/...]. – Desire Kaleba Jun 21 '20 at 8:26

2 no i know about cors , the thing is these are my local files – Shreyan Mehta Jun 23 '20 at 19:10

3 You need to serve your script in an http server, browsers use an http request to load es6 modules, the server needs to respond in the header a CORS allowing your origin. – danilo Jul 8 '20 at 1:22

1 the simplest way: you can use http-server: stackoverflow.com/a/23122981/935330 – danilo Jul 8 '20 at 1:39

35

I solved this issue by doing the following:

When using ECMAScript 6 modules from the browser, use the .js extension in your files and in the script tag add `type = "module"` .

When using ECMAScript 6 modules from a Node.js environment, use the extension `.mjs` in your files and use this command to run the file:

```
node --experimental-modules filename.mjs
```

Edit: This was written when node12 was the latest LTS, this does not apply on node 14 LTS.

Share Improve this answer Follow

edited Apr 11 at 17:24

answered Feb 5 '20 at 9:56

Anurag Phadnis

509 4 10

2 *Modules: ECMAScript modules: Enabling* `https://nodejs.org/api/esm.html#esm_enabling` – noobninja Sep 13 '20 at 21:41

This is no longer necessary. Simply add "type": "module" to your package.json and everything will work as expected. (Use .js for filename extensions) – Chris Perry Apr 9 at 19:20

25

I resolved my case by replacing "import" by "require".

```
// import { parse } from 'node-html-parser';
parse = require('node-html-parser');
```

Share Improve this answer Follow

answered May 15 '20 at 0:22

nik s

277 3 2

thanks, I solved mine with this syntax as well – IdontEvenEven 2 days ago

I don't know whether this has appeared obvious here. I would like to point out that as far as client-side (browser) JavaScript is concerned, you can add `type "module"` to both external as well as internal is

You can use it in an external script, in which you do the import, eg.:

```
<!DOCTYPE html><html><body>
<script type="module" src="test.js"></script><!-- Here use type="module" rather
than type="text/javascript" -->
</body></html>
```

test.js:

```
import {a} from "./module.js";
alert(a);
```

You can also use it in an internal script, eg.:

```
<!DOCTYPE html><html><body>
<script type="module">
    import {a} from "./module.js";
    alert(a);
</script>
</body></html>
```

It is worthwhile mentioning that for relative paths, you must not omit the "./" characters, ie.:

```
import {a} from "module.js";     // this won't work
```

Share Improve this answer Follow

edited Nov 20 '20 at 12:20

Fury
4,193 4 40 71

answered Jul 25 '20 at 7:00

Chong Lip Phang
6,640 5 50 74

13

For me, it was caused before I referred a library (specifically `typeORM`, using the `ormconfig.js` file, under the `entities` key) to the `src` folder, instead of the `dist` folder...

```
   "entities": [
      "src/db/entity/**/*.ts", // Pay attention to "src" and "ts" (this is
wrong)
   ],
```

instead of

```
   "entities": [
      "dist/db/entity/**/*.js", // Pay attention to "dist" and "js" (this is
the correct way)
   ],
```

Share Improve this answer Follow

edited Oct 16 '20 at 3:39

answered Sep 14 '20 at 9:46

4

Adding the why this occurs and more possible cause. A lot of interfaces still do not understand ES6 Javascript syntax/features, hence there is need for Es6 to be compiled to ES5 whenever it is used in any file or project. The possible reasons for the `SyntaxError: Cannot use import statement outside a module` error is you are trying to run the file independently, you are yet to install and set up an Es6 compiler such as Babel or the path of the file in your runscript is wrong/not the compiled file. If you will want to continue without a compiler the best possible solution is to use ES5 syntax which in your case would be `var ms = require(./ms.js);` this can later be updated as appropriate or better still setup your compiler and ensure your file/project is compiled before running and also ensure your run script is running the compiled file usually named dist, build or whatever you named it and the path to the compiled file in your runscript is correct.

Share Improve this answer Follow

edited Jan 5 at 10:50

answered Jan 5 at 4:44

Proxybee

94 1 5

3

The error is triggered because the file you're linking to in your HTML file is the unbundled version of the file. To get the full bundled version you'll have to install it with `npm` :

```
npm install --save milsymbol
```

This downloads the full package to your `node_modules` folder.

You can then access the standalone minified JavaScript file at
`node_modules/milsymbol/dist/milsymbol.js`

You can do this in any directory, and then just copy the below file to your `/src` directory.

Share Improve this answer Follow

edited Mar 8 '20 at 22:38

Peter Mortensen

28.4k 21 95 123

answered Nov 3 '19 at 21:30

rootr

354 1 8

3 `--save` 's been default since `npm 5 & node 8` (2017): stackoverflow.com/q/36022926/1821548, nodejs.dev/npm-dependencies-and-devdependencies, github.com/benmosher/eslint-plugin-import/issues/884, auth0.com/blog/whats-new-in-node8-and-npm5 – Det Nov 8 '19 at 21:30

3

I'm coding on vanilla js. If you're doing same, simply add a type="module" to your script tag.

That is, previous code:

```
<script src="./index.js"></script>
```

Updated Code:

```
<script type="module" src="./index.js"></script>
```

3

1. Go to Project Root Directory `Package.json` file
2. add `"type":"module";`
3. Save it and Restart Server

Share Improve this answer Follow

edited Jun 11 at 10:49

Samsul Islam

2,312 2 13 20

answered Jan 4 at 10:24

Sarthak Raval

101 8

this is not a question just an answer I am not using the question mark it's only a solution for this error facing in react. that's it. any other doubt. – Sarthak Raval Jan 5 at 5:19

this works for vanilla node also – Geoff Langenderfer Jun 11 at 19:31

0

TypeScript, React, index.html

```
//conf.js:
window.bar = "bar";

//index.html
<script type="module" src="./conf.js"></script>

//tsconfig.json
"include": ["typings-custom/**/*.ts"]

//typings-custom/typings.d.ts
declare var bar:string;

//App.tsx
console.log('bar', window.bar);
or
console.log('bar', bar);
```

Share Improve this answer Follow

answered Feb 8 at 11:55

Alex

6,754 26 85 140

0

What I did in my case was to update

```
"lib": [
      "es2020",
      "dom"
    ]
```

with

```
"lib": [
  "es2016",
  "dom"
```

I ran into this error while trying to use import express

0

Instead of `import express from 'express';`

I used `const express = require('express');`

Hope it works!

Share Improve this answer Follow

answered Jun 5 at 12:36

Alalade Samuel

71 3

It's because you haven't exported, the ts file requires a export class format, whereas in js file we would use exports function.

0

So, we have to use `var_name = require("<pathfile>")` to use that file functions.

Share Improve this answer Follow

edited Jun 11 at 10:50

Samsul Islam

2,312 2 13 20

answered Mar 24 at 18:58

John Samuel J

3 2

> Hello... Inspect // Watch ... Your answer is poorly elaborated. It is useful to insert an effective response, with codes and references. Concluding a practical and efficient solution. This platform is not just any forum. We are the largest help and support center for other programmers and developers in the world. Review the terms of the community and learn how to post; – Paulo Boaventura Mar 24 at 19:35

---

for me helped:

0

1. in ts file used: `import prompts from "prompts";`
2. and use `"module": "commonjs"` in tsconfig.json

Share Improve this answer Follow

answered yesterday

Goaul

53 2 7

Just add `.pack` between the name and the extension in the `<script>` tag in src. i.e.:

-2

```
<script src="name.pack.js">
// code here
</script>
```