Tree Shaking In Js

Tree shaking also known as unused exports is a process of removing unused code from Javascript bundles

Why do we need Tree Shaking ?

It is extremely important to keep the production build as light as possible so that it will be downloaded, processed and executed fastes

With emergance of modern Javascript frameworks, the focus of developers are shifted to creating applications in less time and the component-based development architecture has led to creation of multiple files

Each component can have unused code that the other component may not require, or it may not be used anywhere in whole application, but when included it will increase the build size

Ultimately affecting the performance.

For eg →

```
// utility.js
export function foo() {
  Console.log("I am foo");
}

export function bar() {
  console.log("I am bar");
}
```

Now Import the function foo in another file

```
// main.js
import {foo} from 'utility.js'

foo();
```

Assuming that we're here using webpack as a bundler and that too in development mode

In final build you will see something like

```
/* 1 */
/***/ (function (module, __webpack_exports__,
        __webpack_require__) {

  /* unused harmony export bar */
  function bar() {
    console.log("I am bar");
```

```
function foo() {
console.log("I am foo");
}
);
```

Even though we are not using bar function it is included in final build.

It is annotated by comment
/* unused harmany export bar */

stating it is an unused function

How does Tree Shaking Work

Import and export modules introduced in ES6, lead to major breakthrough for Tree shaking.

This is true since 'static' modules are required for tree shaking to function

Before ES6, the dynamic import of CommonJS module was used which allowed importing files conditionally.

```
var module;
if (condition) {
  module = require("foo");
}
else {
  module = require("bar");
}
```

This approach was a hurdle for the Tree Shaking process as it was not possible to decide which module will be imported as import was happening at run time and excluding files at build time was not practical.

When ES6 modules were introduced they implemented static importing, which means all the files had to be imported globally at top.

```
import foo from "foo";
import bar from "bar";
```

This really helped in detecting unused code as it was to determine which code is being used just like modern IDE's and linters does by highlighting unused code

The bundlers like Webpack are so efficient in Tree shaking that they remove almost all the used imports or codes even properties that are exported but not imported anywhere.

```
// person.js
export const Person = {
    name: "Deepa Chaurasia",
    passion: "Blogging"
};

// main.js
import {name} from './person.js';
console.log(name);
```

As property passion is not imported, it will be treated as unused code and will be removed in Tree Shaking.