Module usage

To import three as module the script tag must be of type module.

```
<script type="module">
```

The three framework uses a folder structure. If you take over the complete system, all import statements are correct. If you want to import three.js (`three.module.js`) and a control (`Orbitcontrols.js`), the following results. See https://discourse.threejs.org/t/module-import-usage/11574/18

- `vendor`
  - `three`
    - `build`
      - `three.module.js`
    - `examples`
      - `jsm`
        - `controls`
          - `Orbitcontrols.js`

The collection of basic examples requires only a few modules besides `three.module.js`. Therefore it makes sense to put these modules in a common folder. You can take the current revision of three into account. You simply add to it, and you have about ten three files for one year.

The import

```javascript
import * as THREE from "https://threejs.org/build/three.module.js";
import { OrbitControls } from "https://threejs.org/examples/jsm/controls/OrbitControls.js";
```

then becomes

```javascript
import * as THREE from "./jsm/three.module.120.js";
import { OrbitControls } from "./jsm/OrbitControls.120.js";
```

if you put the modules in your own folder `jsm`. One has a very simple structure.

```
BasicExample
```

```
jsm
```

Now you have to adjust the import path in the modules. But for the few modules this is more clear than creating the complete folder structure.

In module `OrbitControls.120.js`

```javascript
import {
  EventDispatcher,
  MOUSE,
  Quaternion,
  Spherical,
  TOUCH,
  Vector2,
  Vector3
} from "./../build/three.module.js";
```

turns into

```javascript
import {
  EventDispatcher,
  MOUSE,
  Quaternion,
  Spherical,
  TOUCH,
  Vector2,
  Vector3
} from "./jsm/three.module.120.js";
```

That's all!