**Modules in JavaScript**

**Modules: old ways old ways**

**CommonJS and Asysnchronous Module Pattern (AMD)**

Webpack and browserify -> bundles modules into single bundle.js file

**// commonJS // developed for nodes / to include modules**

**var module1 = require(‘module1’);**

var module2 = require(‘module2’);

function fight(){

}

**module.exports = {**

**fight : fight;**

**}**

//AMD was developed for browsers – (not nice ) –old one – it uses require.js – confusing – not native js module

// UMD universal Module system ..

**IIFE** – **Immediately Invoked Function Expression**

(function(){

var harry = 'potter'

var voldemort = " He who must not be named";

function fight(char1,char2){

var attack1 = **Math.floor(Math.random()** \* char1.length);

var attack2 = Math.floor(Math.random() \* char2.length);

return attack1 > attack2 ? `${char1} ` : ` ${char2}` + ' wins';

}

console.log(fight(harry,voldemort));

})();

Pros: // avoiding polluting global name space //reusability

Cons: // still has name clashes // order of the script tag should be correct

**IIFE & Module pattern**

**var fightModule = (function(//it can receive values here){**

var harry = 'potter'

var voldemort = " He who must not be named";

function fight(char1,char2){

var attack1 = Math.floor(Math.random() \* char1.length);

var attack2 = Math.floor(Math.random() \* char2.length);

return attack1 > attack2 ? `${char1} ` : ` ${char2}` + ' wins';

}

**return {**

**fight : fight ,**

**}**

**})(//parameters can be included here);**

console.log(fightModule.fight('harry','voldemort'));

// other pieces of code can access this

// only the things that are needed are exported , others are kept private

// because of closure , it still has access to harry and potter

// this is what jquery does

**ES6 – Modules:** native js

**Script.js**

export function fight(char1,char2){

}

export **default** function jump(){

}

**Import { fight } , jump from ‘./script’**

**// for export default no need to use { } while importing**

**// for export { } should be used for importing**

**// fight or jump now cannot be called in browser console because that is not a global namespace . its in the module**

***Import module1 from ‘module1’;***

***Import module2 from ‘module2’;***

***Export function jump(){***

***}***

**Script.js**

const harry = 'potter'

const voldemort = " He who must not be named";

**export** function fight(char1,char2){

var attack1 = Math.floor(Math.random() \* char1.length);

var attack2 = Math.floor(Math.random() \* char2.length);

return attack1 > attack2 ? `${char1} ` : ` ${char2}` + ' wins';

}

**Index.html**

<script type=”module” src=”./script.js”> </script>

<script type=”module”>

**import { fight } from ‘./script’;**

console.log( fight(‘ron’,’von’));

</script>

Modules should be served from a server so type

**live-server in the terminal (js file location) and browser shows the result**