ES 2015+

NATIVE MODULES ON BROWSERS

AUTHOR: AMPM

2018

CONTENT

1. What
2. Why
3. Now
4. Support
5. How
6. Demo
7. Resources

WHAT

WHAT

Module is:

- group of functionalities
- Which you exposed all or some of it
- can be based on functions
- can be based on object
- can be based on class syntax
  (of es2015 syntax sugar)

- it is a Pattern
- it is reusable
- it is maintainable
- by using one or group of files

more about:

https://medium.freecodecamp.org/javascript-modules-a-beginner-s-guide-783f7d7a5fcc

WHY

WHY

In the past, to write modules on es2015 syntax you need or use one of this:

- Babeljs (transpiler)
- Traceur (transpiler)
- Typescript (superset of JS + transpiler)
- SystemJS (Module Loader)

And to use previous tools and be more productive you use one of this:

- Grunt (Task runner)
- GulpJS (Task runner)
- Browserify(Module Bundler)
- WebPack (Module Bundler)
- Rollup (Module Bundler)
- ParcelJS (Module Bundler)

NOW

(since last semester of 2017 or last 3 month of 2017)

- You don't need tools like transpilers or a temporary module loader
- because After 2 years of waiting and despairing

- The majority of the browser (or at least the moderns) support es2015 module !!!
- And use the tools like Webpack or Parcel, only to minify(js, css)

SUPPORT

SUPPORT

Browser:

Safari: since 10.1 version

Chrome: since 61 version

Opera: since 48 version

Firefox: since 54 version, behind the dom.moduleScripts.enabled setting in about:config

Edge: since 15 version, behind the Experimental JavaScript Features setting in about:flags

HOW

HOW - THE MODULE

create a js file that will be a module - first way

```
// module.js
// first way

function ab (){ }

function foo (){ }

function bar (){ {}

// make visible only foo and bar function
export {foo, bar}
```

create a js file that will be a module - second way

```
// module.js
// second way

function ab (){ }

//make visible
export function foo (){ }

//make visible
export function bar (){ {}
```

HOW - CALL IT

create main.js file - first way

```
// main.js
// select which functionality you want
import {bar, foo} from './module.js';

let b = bar();

let f = foo();
```

note:

create main.js file - second way

```
 // main.js
// create an alias for the module and load all function
// that are visible
 import * as myModule from './module.js';

 let b = myModule.bar();

 let f = myModule.foo();
```

note:

don´t forget to add the js extension on name of module

HOW - LOAD IT

create the index.html file

```
(...)

<!-- between the body tag -->
 <script type="module" src="main.js"></script>

 (...)
```

DEMO

RESOURCES

RESOURCES

Go to:

https://github.com/ampmonteiro/es2015-native-module-loader

(see below video description)

SLIDES & VIDEO WITH

- QuickTime (Screen Record)
- RevealJs (v.3.6.0)
  https://revealjs.com

THANKS FOR WATCHING ;)

For more videos, search on youtube:

jediampm